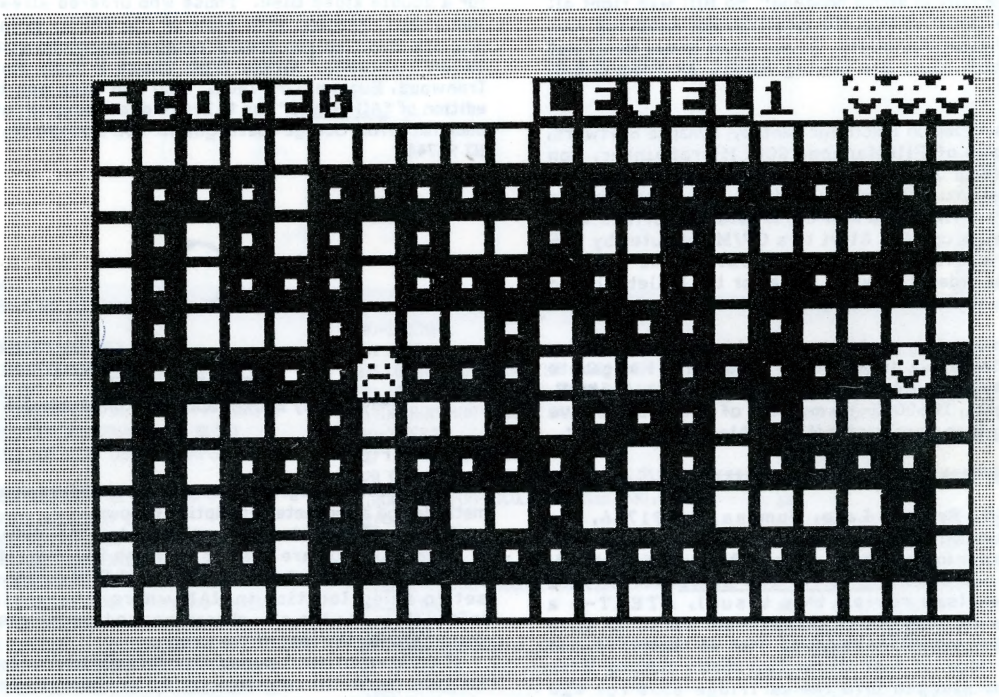
ATARI
COMPUTER
ENTHUSIASTS
3662 Vine Maple Dr. Eugene OR 97405

Aug/Sept 1982
Mike Dunn and Jim Bumpas, Editors



Gobbler
by Sydney Brown

program inside

News Bits

The July 26, 1982 edition of InfoWorld ($25 yr, call 800-343-6474) is devoted to the Atari computer and has all kinds of interesting tidbits. In case you are not familiar with InfoWorld, it is a weekly newspaper for the Microcomputer, and is very worth while if you are interested in the latest news, although Marc Benioff is usually even more current!

They begin by saying that about 300,000 Ataris have been sold and that Atari is outselling Apple. They report that Roger Baderscher, the former head of Atari, is leaving to form his own company. There is sort of a panel discussion with Bill Wilkinson of O.S.S., Ken Grant from Synapse, Robin Sherer from Santa Cruz Software, and Jim Capparell of Antic with Cylde Spencer from the San Francisco Bay Area Atari Group representing the user's. Some interesting comments on the Atari BASIC vs Microsoft's were that now that there are so many Atari programs and books on programming in Atari BASIC are out, the lack of compatability with Microsoft has become much less inportant. As Bill Wilkinson points out, the new ANSI standards for the new BASIC, as reported in a recent issue of Byte, show Atari BASIC to be much closer than Microsoft, especially in string handling!! So Bill was right all along when he insisted that the Atari String handling was superior to that of MicroSoft. In our local area, now that MicroSoft is out, many are buying BASIC A+ from O.S.S. instead!!

There are articles on Electronic Fantasy, Synapse Software, and nice reviews of FileManager 800, MicroPainter, Lon Poole's book, "Your Atari Computer" and Monkey Wrench, all similiar to those you have already read in A.C.E. There is also an interesting article on a new CP/M-Atari Interface that allows you to hook up your Atari to a CP/M computer by USS for $177, and a review of the new PERCOM double-density drives- we have ordered a double drive for the bulletin board and will review it and the Leading Edge drive soon.

The most interesting product announced in this issue was a 3 inch Micro-FloppyDisk drive and cartridge with 1 megabyte capacity for $899 by Amdek, 2420 E. Oakton St, Suite E, Arlington Heights, Il 60005- no mention of Atari, but I have heard that they plan on releasing it in an Atari version "soon".

****

ELCOMP 53 Redrock Lane, Pomona, CA 91766, the German-based computer company will announce at the London Personal Computer in September some of their new products, including the new book, Games for the Atari ($7.95, with cassette, $40)(see review this issue), ATEXT-1 a wordprocessor for only $30 on cassette but works for both cassette and disk and will be reviewed when we receive it, and a EPROM-cartridge kit that allows two 2532 , 2732, or 2716 EPROMS to be plugged into the cartridge slot for $20 (bareboard) or $30(with EPROMs). They also have an EPROM maker ready that uses the joystick ports but I don't know the price yet.

Consumer Reports is looking for information for user's and their experiences with personal computers, presumably for an upcoming issue. They have asked for information about all the "popular" computers, specifically, Apple, Pet, IBM and Radio Shack, but left out Atari!! Please tell them about your computer, so we are not left out. Write to Consumer Reports, Box RME, 256 Washington St., Mt. Vernon, NY 10550.

Several companies are releasing Trac Balls now- will keep you updated.

-Mike Dunn

August Meeting

Meetings are always on the 2nd Weds night at 7:30, except this month there will be no regular meeting. Instead, there will be a gala pot-luck picnic at the covered picnic tables at Hendrick's Park on Wednesday, August the 18th. Call Kirt Stockwell on details and what to bring. There is power for computers at the picnic site.

Bulletin Board
(503) 343-4352

We have obtained a used Atari 400 for the dedicated Board that will run 24 hours a day. A new Canadian company, Tara,3648 southwestern Blvd., Orchard Park, N.Y. 14127 or 2 Robert Speck Parkway, Suite 1500-S, Mississauga, Ontario L47-1H8 Canada, has kindly donated a 48K board for the 400 that retails for $199!! When we receive it, we will install it, review it and you can see it in action. We have also ordered a dual double-density Percom disk drive, that will eliminate many of the problems that we have been having with using only one drive- we will be able to put much more on line, and still have room for your messages, programs that you upload, etc. Frank Hubband has upgraded the Armudic Bulletin Board to work with the Percom drives and has many new features as well as fixing of the bugs- will go on line when we receive it. By the end of August all should be up- Kirt Stockwell will be SYSOP for the last week or so of August.

Remember, you can obtain "Best of Ace-1982 1/2" Game Disk or Cassette, or Utility Disk and support the Bulletin Board, as well as get some great programs for $15 each or $20 for a double sided Disk. Those who ordered already- there was a slight problem with some of the Utility programs, but now are fixed so you should receive you disks soon, if you haven't already. Order from Chuck & Jody Ross, 2222 Ironwood, Eugene, Or 97401. You can also get the bound edition of "All of ACE" for $12-$14 depending on !st class or book rate from George Suetsugu, 45-602 Apuapu St, Kaneohe, HI 96744.

GOBBLER

-by Sydney Brown, Rockhampton, Australia

(See cover picture)

The Gobbler program uses mostly standard programming methods and a complete description follows.

Lines 0-100 are standard and just put up the instructions.

Lines 100-105 are used to redefine the character set by redirecting the computer from the normal ROM based character set to a new location in RAM where I have placed the information for the new characters and copied the original characters for the rest of the area.

I have reserved 2 pages (512 BYTES) at the top of user RAM and placed the new character set up there, by reading the data statements in Lines 1000-1009 and placing them in the correct position and order. Each number represents the binary code for one line of one character.

Lines 190-199 are used to initialize all the variables and also modifies the basic maze to the level you have reached.

Lines 250-299 determine how many Globs are allowed at the level you have reached.

Lines 300-389 are used to give the Globs their limited intelligence and control their movement.

Lines 400-410 are used to produce the smiles on the Globs and the shocked look on the man.

Instead of having extra characters I just POKEd the slight variation directly into the new character set to modify the correct positions. This changes all the characters simply and quickly without having to PLOT the new characters into the positions occupied by the Globs or the men.

Lines 500-599 are used in connection with 600-699 to display the end of a level and stepping up to the next level.

Lines 600-658 are used to see when the man hits a nugget, a bag of gold, or goes through the tunnel at the edge of the screen.

Lines 900-919 draws the maze and the boundaries for each level.

Lines 920-999 are used to fill all the blank areas with nuggets.

H. Brown, Australia

Add 221 - Change 220 - Add STR$ - Add TRAP to 200

GOBBLER

Add 920 ⇒

ADD line 221 IF EX=0 THEN IF 2<7 32 THEN COLOR 2: PL. 1+,V.

235 = 6 32 = space 234 = man 172 = walls

```
6 REM xxxxxxxxxxxxxxxxxxxxxx
7 REM xx     GOBBLER     xx
8 REM xx        by       xx
9 REM xx  Sydney Brown    xx
10 REM xx    Aug/Sept 82  xx
11 REM xxxxxxxxxxxxxxxxxxxxxx
15 ? "ↄ":DIM C(5),H(5),V(5)
20 POKE 710,224:POKE 709,14:? "
":? "     |    G O B B L E
 R    |"
22 POKE 752,1:? "
":? :? "YOU ARE INSIDE A MAZE COLLECTING G
OLD"
23 ? "NUGGETS,BUT YOU HAVE A PROBLEM.":? "THE
 GOLD YOU ARE COLLECTING IS NOT":? "YOURS,IT
BELONGS TO THE GLOBS."
25 ? "AS YOU CAN IMAGINE THE GLOBS ARE NOT":?
 "AT ALL PLEASED AND TRY TO CATCH YOU."
27 ? "YOU TRY TO COLLECT ALL THE GOLD YOU":?
"CAN BEFORE YOU ARE CAUGHT !!!":? :? "EACH LE
VEL IS MORE DIFFICULT"
28 ? "YOU HAVE 4 CHANCES TO COLLECT THE GOLD"
;
29 ? "USE JOYSTICK 1 TO CONTROL YOUR SMILEY":
? "EACH NUGGET SCORES 1 POINT & THE BAG":? "O
F GOLD SCORES 20"
32 ? :? "A BONUS OF 100 FOR A COMPLETED MAZE"
:? :? "by Sydney Brown-Fun-Co  ":? "given to
ACE members"
40 ? "Press START to start the game";:POKE 53
279,0
50 IF PEEK(53279)<>6 THEN 50
100 POKE 106,PEEK(106)-2:GRAPHICS 18:? #6;"
   setting up"
101 POKE 710,140:POKE 708,14:POKE 709,236:POK
E 711,222:A=PEEK(106)*256:FOR B=0 TO 511
102 IF B>319 AND B<352 THEN READ D:POKE A+B,D
:NEXT B
103 IF B>367 AND B<376 OR B>383 AND B<392 THE
N READ D:POKE A+B,D:NEXT B
105 ZZ=PEEK(57344+B):POKE A+B,ZZ:SOUND 0,ZZ,1
0,8:NEXT B:POKE 756,PEEK(106)
190 X=19:Y=0:AX=0:L=1:S=0:POSITION 0,0:? #6;"
score0    LEVEL1jjjj":REM "score and "level"
and "jjjj" in inverse
194 COLOR 234:PLOT X,Y:W=9:V=0:ND=0
195 ON L GOSUB 900,920,900,920,900,920,900,92
0,900,920,900,920,900,920,900,920,900,920,900
,920,900,920,900,920
196 COLOR 32:PLOT 16+AX,0:POSITION 15,0:? #6;
L;:IF L>10 AND L/2<>INT(L/2) THEN COLOR 72:PL
OT 0,6:PLOT 19,6:ND=2
197 EX=22:X=18:Y=6:CY=1:COLOR 234:PLOT X,Y:IF
 L>12 THEN COLOR 72:PLOT 10,3:PLOT 10,9:ND=ND
+2
198 H(1)=1:V(1)=6:C(1)=235:H(2)=10:V(2)=10:C(
2)=235:H(3)=6:V(3)=2:C(3)=235:IF L>14 THEN CO
LOR 72:PLOT 2,6:ND=ND+1
199 H(4)=18:V(4)=3:C(4)=235:H(5)=9:V(5)=6:C(5
)=235:IF L>16 THEN COLOR 72:PLOT 14,6:ND=ND+1
200 ST=STICK(0)
201 IF ST=11 AND X<1 THEN LOCATE 19,Y,Z:IF Z=
32 OR Z=235 OR Z=112 THEN GOSUB 600:X=X-1:IF
X<0 THEN X=19
202 IF ST=11 THEN LOCATE X-1,Y,Z:IF Z=32 OR Z
=235 OR Z=112 THEN GOSUB 600:X=X-1:IF X<0 THE
N X=19
203 IF ST=7 AND X>18 THEN LOCATE 0,Y,Z:IF Z=3
2 OR Z=235 OR Z=112 THEN GOSUB 600:X=X+1:IF X
>19 THEN X=0
204 IF ST=7 THEN LOCATE X+1,Y,Z:IF Z=32 OR Z=
235 OR Z=112 THEN GOSUB 600:X=X+1:IF X>19 THE
N X=0
206 IF ST=14 THEN LOCATE X,Y-1,Z:IF Z=32 OR Z
=235 OR Z=112 THEN GOSUB 600:Y=Y-1
208 IF ST=13 THEN LOCATE X,Y+1,Z:IF Z=32 OR Z
=235 OR Z=112 THEN GOSUB 600:Y=Y+1
209 IF ST<>15 THEN COLOR 234:PLOT X,Y
220 EX=EX+1:IF EX>100 THEN H=INT(RND(0)*17)+1
:V=INT(RND(0)*9)+1:LOCATE H,V,Z:IF Z=32 THEN
COLOR 112:PLOT H,V:EX=0
222 IF EX=20 THEN COLOR 32:PLOT H,V:COLOR 234
:PLOT X,Y:POKE 77,0
250 CY=CY+1:IF CY>5 THEN CY=1
251 ON L GOTO 268,268,266,266,264,264,262,262
,260,260,260,260,260,260,260,260
260 IF CY=5 THEN N=5:GOSUB 300:GOTO 200
262 IF CY=4 THEN N=4:GOSUB 300:GOTO 299
264 IF CY=3 THEN N=3:GOSUB 300:GOTO 299
266 IF CY=2 THEN N=2:GOSUB 300:GOTO 292
268 IF CY=1 THEN N=1:GOSUB 300:GOTO 290
290 IF L=1 OR L=2 THEN FOR W=1 TO 10:NEXT W
292 IF L=3 OR L=4 THEN FOR W=1 TO 5:NEXT W
299 GOTO 200
300 IF Y>V(N) THEN 350
301 IF X<H(N) THEN 325
302 IF Y<V(N) THEN 375
303 LOCATE H(N)+1,V(N),Z:IF Z=234 THEN 400
304 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):H(N)=H(N)+1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
305 LOCATE H(N),V(N)+1,Z
306 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):V(N)=V(N)+1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
307 LOCATE H(N),V(N)-1,Z
308 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):V(N)=V(N)-1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
309 RETURN
325 LOCATE H(N)-1,V(N),Z:IF Z=234 THEN 400
326 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):H(N)=H(N)-1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
327 LOCATE H(N),V(N)+1,Z
328 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):V(N)=V(N)+1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
329 LOCATE H(N),V(N)-1,Z
330 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):V(N)=V(N)-1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
339 RETURN
350 LOCATE H(N),V(N)+1,Z:IF Z=234 THEN 400
351 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):V(N)=V(N)+1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
352 IF H(N)<19 THEN LOCATE H(N)+1,V(N),Z
353 IF (Z=32 OR Z=235) AND H(N)<19 THEN COLOR
 C(N):PLOT H(N),V(N):H(N)=H(N)+1:COLOR 201:PL
OT H(N),V(N):C(N)=Z:RETURN
354 LOCATE H(N)-1,V(N),Z
355 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):H(N)=H(N)-1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
369 RETURN
375 LOCATE H(N),V(N)-1,Z:IF Z=234 THEN 400
376 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):V(N)=V(N)-1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
377 LOCATE H(N)+1,V(N),Z
378 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):H(N)=H(N)+1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
379 LOCATE H(N)-1,V(N),Z
380 IF Z=32 OR Z=235 THEN COLOR C(N):PLOT H(N
),V(N):H(N)=H(N)-1:COLOR 201:PLOT H(N),V(N):C
(N)=Z:RETURN
389 RETURN
400 PP=PEEK(106)*256+332:POKE PP,189:POKE PP+
1,195:POKE PP+8,231:FOR W=0 TO 160 STEP 10:FO
R WW=W TO W+20 STEP 2
401 SOUND 0,WW,10,10:IF WW>W+9 THEN COLOR 106
:PLOT X,Y
402 IF WW<W+10 THEN COLOR 234:PLOT X,Y
404 NEXT WW:NEXT W:AX=AX+1
405 COLOR 238:PLOT X,Y:FOR W=250 TO 5 STEP -7
:SOUND 0,W,2,14:NEXT W:COLOR 32:PLOT X,Y
408 POKE PP,195:POKE PP+1,189:POKE PP+8,153:S
OUND 0,0,0,0
409 IF AX<4 THEN 194
410 COLOR 238:PLOT X,Y:POSITION 4,11:? #6;"ga
meHHHHover":POKE 53279,0
412 FOR W=1 TO 100:NEXT W:FOR WW=1 TO 3:FOR W
=0 TO 255 STEP 2:SOUND 0,W,10,10:SOUND 1,255-
W,10,10:NEXT W:NEXT WW
415 SOUND 0,0,0,0:SOUND 1,0,0,0
420 IF PEEK(53279)<>6 THEN 420
421 POSITION 0,0:? #6;"ↄ":GOTO 190
500 L=L+1:POSITION 15,0:? #6;L
599 GOTO 194
600 COLOR 32:PLOT X,Y:IF Z=32 THEN RETURN
605 IF Z=112 THEN 620
610 SOUND 0,49,10,10:S=S+1:ND=ND+1:POSITION 5
,0:? #6;S:IF L/2<>INT(L/2) AND ND>105 THEN 65
0
612 IF L/2=INT(L/2) AND ND>160 THEN 650
619 SOUND 0,0,0,0:RETURN
620 S=S+20:POSITION 5,0:? #6;S:FOR WW=1 TO 7:
FOR W=0 TO 50 STEP 10:SOUND 0,W,10,14
622 IF WW=1 OR WW=3 OR WW=5 OR WW=7 THEN COLO
R 234:PLOT H,V
623 IF WW=2 OR WW=4 OR WW=6 THEN COLOR 112:PL
OT H,V
629 NEXT W:NEXT WW:SOUND 0,0,0,0:RETURN
650 LOCATE X,Y+1,Z:IF Z=235 THEN Y=Y+1:GOTO 6
60
652 LOCATE X,Y-1,Z:IF Z=235 THEN Y=Y-1:GOTO 6
60
654 IF X<19 THEN LOCATE X+1,Y,Z:IF Z=235 THEN
 X=X+1:GOTO 660
655 IF X>0 THEN LOCATE X-1,Y,Z:IF Z=235 THEN
X=X-1:GOTO 660
657 LOCATE 0,6,Z:IF Z=235 THEN X=0:Y=6:GOTO 6
60
658 LOCATE 19,6,Z:IF Z=235 THEN X=19:Y=6
660 COLOR 234:PLOT X,Y:FOR W=190 TO 0 STEP -5
:SOUND 0,W,10,10:SOUND 1,W+1,10,10:IF W/10=IN
T(W/10) THEN POKE 712,184
662 IF (W-5)/10=INT((W-5)/10) THEN POKE 712,0
664 NEXT W:S=S+100:POSITION 5,0:? #6;S
699 SOUND 0,0,0,0:SOUND 1,0,0,0:POKE 712,0:GO
TO 500
900 COLOR 72:PLOT 0,1:DRAWTO 19,1:DRAWTO 19,1
1:DRAWTO 0,11:DRAWTO 0,1:PLOT 4,2:PLOT 4,3:PL
OT 4,10:PLOT 4,9
901 PLOT 2,3:DRAWTO 2,5:DRAWTO 4,5:PLOT 2,9:D
RAWTO 2,7:DRAWTO 4,7:PLOT 8,3:PLOT 9,3:PLOT 8
,9:PLOT 9,9:PLOT 6,3
902 DRAWTO 6,5:DRAWTO 8,5:PLOT 6,9:DRAWTO 6,7
:DRAWTO 8,7:PLOT 11,3:PLOT 12,3:PLOT 12,4:PLO
T 11,9:PLOT 12,9
903 PLOT 12,8:PLOT 10,5:PLOT 10,7:PLOT 10,6:D
RAWTO 12,6:PLOT 17,3:DRAWTO 14,3:DRAWTO 14,5:
PLOT 14,7:DRAWTO 14,9
904 DRAWTO 17,9:PLOT 16,5:DRAWTO 18,5:PLOT 16
,7:DRAWTO 18,7
910 COLOR 235:FOR X=1 TO 18:FOR Y=2 TO 10:LOC
ATE X,Y,Z:IF Z=32 OR Z>72 THEN PLOT X,Y
911 NEXT Y:NEXT X:PLOT 19,6:PLOT 0,6:COLOR 23
4:PLOT 18,6
919 RETURN
920 COLOR 235:FOR X=1 TO 18:FOR Y=2 TO 10:PLO
T X,Y:NEXT Y:NEXT X:COLOR 72:PLOT 0,6:PLOT 19
,6
999 RETURN
1000 DATA 255,129,129,129,129,129,129,255
1001 DATA 60,126,219,255,195,189,255,170
1002 DATA 60,126,219,255,153,195,102,60
1003 DATA 0,0,0,24,24,0,0,0
1004 DATA 145,82,20,120,27,156,42,65
1009 DATA 8,12,14,56,124,254,254,124
```

Bags of gold

108

New Maze - 920 ⇒

7,1
14 8,1
2,2
3,2
4,2
5,2
7,2
9,2
10,2
12,2
14,2
16,2
18,2
4,3
12,3
16,3
3,4
4,4
6,4
7,4
8,4
9,4
11,4
12,4
13,4
15,4
16,4
17,4
8,5
1,6
2,6
3,6
5,6
6,6

14,6
16,6
18,6
5,7
11,7
2,8
3,8
5,8
10,8
11,8
13,8
14,8
16,8
18,8
7,9
15,9

45 RESTORE 1000

92 READ X, Y:

924 COLOR 72: PL
IF W<>54 THEN

920 RESTORE 920: L

926 COLOR 235: FO
TO 18: FOR Y=2 TO 10: LO
X,Y,2: D$=STR$(0): IF
OR Z>72 THEN PLOT X,

928 NEXT Y: NEXT X:
19,6: PLOT 0,6: COLOR 2
PLOT 18,6

COLOR 72
929 PLOT 0,1: DR
DR 19,11: DR. 0,11:

599 GOTO 190

190 X=19: Y=0: ? #6; "clear"; POS.
? #6; "score ....."
140 L=1: S=0: AX=0

REVIEWS

Page Six

(Synapase, 820 Coventry Rd., Kensington, CA 94707 $30)

Page Six is a collection of useful utilities that you may add to your own programs. The disk is copyrighted, but you may incorporate any of the routines in your own programs and do what you like with them. Most of the utiltiies are located on page 6, and include both the assembly language listing as well as the BASIC listing for your use.

Programs include a joystick expander program to simplify joystick programming and an "informer" program that allows you to see the status of various flags such as cursor column and row, keyboard buffer, etc. There are programs that allow you to print banners 1 1/2 inch high either vertically or horizonally, a slow list utility to allow you to control the speed of a listing and a program that allows you to better utilize the console switches on your Atari.

Some of the more useful utilties on the disk is a Mini-DOS that resides in memory on top of BASIC, A "Musician" program that allows you to write short musical piece easily by spcifying the note, octave, tempo, etc in musical notations, and a Textual Display Enhancer that allow 4 different colors to be displayed in Graphics 0.

All in all, many useful programs , written by sometime ACE contributor Matt Loveless and one of his friends. Our "Best of 1982 1/2 utility disk" uses the text enhancer for the title page.

-Mike Dunn

Atari Games and Recreations

by Herb Kohl, Ted Kahn, Len Lindsey)
(Reston Publishing Co. Reston, VA)

Here it is! A large, 338 book that teaches you to program in Atari BASIC through Games and fun. Very well written and entertaining, the book covers BASIC programming but also all the special features of the Atari, including Graphics, animation, sound and music, and color. There are lots of programs listed, all fully explained. A book that I could recommand to all most highly!!

-Mike Dunn

Instant BASIC- 2nd Astounding! Edition

by Jerald Brown

(dilithium Press, POB 606, Beaverton, OR 97075 $12.95)

This book uses a rather unusual approach to learing BASIC. Each page is arranged differently and each has a variety of pictures, drawings, symbols, and instructions. This variety may be an attempt to make it light and interesting but was distacting to me and made it difficult to concentrate and move through the book smoothly.

The book must be used like a workbook and cannot be used as a reference book. The greatest drawback is problably the fact that it is not written just for the Atari computer but for many microcomputer BASIC's. Differences between computers are frequently mentioned and may be confusing to some people.

While the approach of this book is usiguq and may be very helpful to some people, I found the book "Your Atari Computer",reviewed last month, the most organized and succinct for learning "Atari BASIC" as a beginner.

-Ken Springate

note from Editor: I rather liked the approach of this book, the graphics and the comparisons of the various BASIC's. The three books mentioned above are all good for the beginner- "Your Atari Computer" is best on teaching you to use your Atari, especially if you just got it. Atari Games and Recreation is an excellent guide to learing Atari BASIC, and I like Instant BASIC as a general beginner's guide to BASIC with comparisons with other popular BASIC's if you are interested in learning to translate. You can't go wrong with any of them.

-Mike Dunn

GAMES FOR THE ATARI

By S. Roberts
(ELCOMP, 53 Redrock Lane, Pomona, CA 91766 $7.98, or $40 with cassette)

What does GAMES FOR THE ATARI cover? It covers Display List Interrupts, GTIA or CTIA chip, Player Missile Graphics, Redefining the Character Set, Sound, and Movement in Basic and Machine language; GAMES FOR THE ATARI goes into good depth. It has most of the information the intermediate programer needs to know.

How does it go over the material? First it tells a brief summary of what it is explaining, second it goes over examples it shows through diagrams or through programs, next it tells what lines in the program serve as what function.

What type of examples or games are in the book? There are many different types of programs but for the most part they are games; here are some titles Backgammon, Bomber, Gunfighter, Calender, Robot-Attack, and Barrier.

My opinion of GAMES FOR THE ATARI. I feel that it gave more understanding and more structured programing for the Atari. I recommend this book for the intermediate programer who don't like most books because they are boring and don't give a lot of understanding. I have really enjoyed doing this review for the Eugene Atari Computer Enthusiasts (ACE)

-L.J. Knoll

PROM-IT

EPROM development system for the Atari 800 computer by MPC Peripherals Corporation, 9424 Chesapeake Dr., San Diego, CA 92123)

What is an EPROM? An EPROM is an acronym for Eraseable Programmable Read Only Memory. It is eraseable by exposure to ultraviolet light. An EPROM works like a ROM but can be reused and reprogrammed. This package consist of both the hardware and software. The hardware connects to controller ports 3 and 4 of the computer. The software has a good menu and allows you to Check for erasure, Verify against RAM buffer, Burn from RAM or disk file, Copy another EPROM, Save to disk file, Move to or Load from RAM buffer, Display EPROM to screen or printer. It also comes with Personality Modules so that you can burn 2508, 2516, 2532, 2716, and 2732 EPROMs.

The package is complete and very easy to use and it works very well, so far I have had no problems in any way with it. At $199 from MPC it is worth it. Lets see more good products.

-E.J. Knoll

FULL-VIEW 80

Hardware review of the 80 column display board by Bit 3 Computer Corp., 8120 Penn Ave., Minneapolis, MN $349.

by David Stellmack
Technical Editor
ACE of Columbus

(This is a new Atari Newsletter and with Dave as it's technical editor, should have some great articles. Dave kindly sent me this review, which will also be published in his newsletter. Contact him at 4615 Healy DR., Columbus, Ohio 43227.)

The Full-View 80 is the first 80 column board for the Atari. Why get an 80 column board? Well, I have a special reason. The computer I use at work requires an 80 column terminal and I want to use that computer via modem from home. So, I rushed right out and got a Bit 3. I'm sorry to say I wish I had waited.

First you must understand the Bit 3's system requirements. The Full-View 80 is available only for the Atari 800. A 32k RAM board must be in the 2d slot of the memory area, and a 16k board in te 1st slot. The last empty slot in the memory area is where the Full-View 80 resides. I was lucky. I have 48k with a 32k RAM CRAM. A monitor is also required.

This is where trouble began for me. I have a NEC composite color monitor. I purchased the color monitor because I knew the 80 column boards were coming. Well, I didn't see in the Bit 3 literature the following line: "Video monitor is required -- color TV sets or most color monitors cannot provide a satisfactory 80 column display."

Well, I was in a real fix. I went to my friend who has an Apple and used his Amdek 300 hi-res green screen to finish this review.

The Full-View 80 provides an 80x24 display which looks very good. The lower case letters have true descenders. The character size is 8x10 cell which looks very good and easy to read. The EPROM on the board provides for 128 characters and the documentation says the EPROM can be reprogrammed to allow the user to make character sets. The standard set contains upper and lower case ASCII characters plus line drawing graphics suitable for business forms.

The on board firmware in ROM make the board usable from BASIC or machine language. The Full-View 80 does not use any of the ATARI RAM to display the screen. A software controlled video switch permits one to switch between the ATARI 40x24 and 80x24 under keyboard or program control.

In BASIC to go to 80x24 type: A=USR(54818) <RETURN>. All normal commands such as List, Print, Run, Break, Tab, Insert Char, Delete Char, DOS, and the rest are supported in 80 column format. Use Open and Close statements to use the Full-View 80 in a program.

The board comes with 60 Hz operation standard and 50 Hz firmware is available by request. 80 column word processing will be supported with Full-View 80 and the new 80 column Letter Perfect from LJK. An 80 column assembler called Edit 6502 and an 80 column data base manager called Data Perfect will be available from LJK.

Well, I guess it's time to rate the board on a scale from 1-10 (10 the best). I give the Full-View 80 a 6 because of the non-compatibility of the NEC monitor. If you have a nice non-color monitor this board is for you. If not, you are still waiting like I am.

NOTE:

I took the Full-View 80 back and got my money back because I want an 80 column board which I can use with the NEC color monitor. I also called Bit 3 and spoke to a tech rep. The rep told me the following: The problem is not a new one and is general with all boards and personal computers (even the Apple). Amdek has a converter to improve the Full-View 80 with the Apple but does not have any such converter for the Atari. The following software is compatible with the Atari: 8k BASIC, Atari Microsoft BASIC, BASIC A+, DOS 2.OS, Editor Assembler, APX Pascal, and many of the other common packages. The ones they know which do not work are: Medit, Telelink, and many of the direct cursor addressing programs, or programs which invoke a different graphics mode. LJK plans to release 3 programs compatible with the Full-View 80, Atari version.

Hints

by Matt Giwer

Some months ago the newsletter published a question on how to use the features of the ATARI Word Processor with the EPSON printer. Perhaps I don't understand the question but it is rather simple. To print something in italics, for example, the key sequence is CONTROL/INSERT, ESC, 4 and you will get this effect. To turn it off you use the same sequence except 5 instead of 4 and the italics go away. The other double strike and so forth are the same. The CONT/INSERT [INSERT is the shifted >] permits the ESC code to go into the text rather than the software shifting modes. To make a direct translation to some of the more complex functions of each it will be necessary to add the GRAFTRAX chips at $60 to $90 and then redefine the control codes of the EPSON to those of the WP. This I have not had the interest to go into. The other approach is to go into the WP software and change the jump table for the control codes to put EPSON codes in the text. That seems to be rather more effort than it is worth. If this is the answer to the question I regret not writing earlier.

Inside Atari DOS

by Bill Willkinson

(Compute! Books, POB 5406, Greensboro, NC 27403, (800)334-0868, $20)

This book was written by Bill Willkinson of O.S.S., the authors of Atari BASIC, BASIC A+, and the Atari DOS. Bill is also the author of the series of articles in Compute! called Insight:Atari that is must reading for serious Atari owners. Beginning with a fascinating chapter on the history of the Atari computer and Atari BASIC, it goes on to an overview of the Atari DOS. The book is about the File Manager System part of DOS 2S, that is, the file called DOS.SYS. The DUP.SYS part is owned by Atari and the listings can be obtained from them. O.S.S. owns the rights to the DOS.SYS. After the overview of the DOS, each command and procedure used by the DOS is explained in exhaustive detail. The final part of the book is the complete, well-documentated assembly-language listing of the complete DOS.SYS.

If you would like to know all about how DOS 2S works, this is the book for you. Assumes a fair amount of knowlege of assembly language; not for beginners.

Affordable Modems

The Anchor Direct Connect Modem called the Signalman Mark II is now available in an Atari version that plugs right into the interface and costs only $99. It works with the modular type phone only, connecting directly into the headset. It works well, and with the Jonestrm modem program in the June issue (also in the BEST of ... above), you have a very inexpensive way to enter the world of computer communications. I have tested the two together and they work fine.

Microbits (434 W. 1st St., Albany, Or 97321 (503)967-9075, who are local members of ACE are marketing a modified version of above for Atari owners without an interface. Their modem will connect directly into the joystick ports and come with special terminal software. They have given the club several prototypes and they work fine. When they are nationally marketed, they will retail for $160, but ACE members can get them directly from Microbits for $140.

ADD A LITTLE JOY

USING THE JOYSTICK AND PADDLE IN ATARI PILOT

by Ruth Ellsworth

The simplicity of ATARI PILOT makes is easy to use the joystick or paddle to make programs in PILOT more interesting and enjoyable for children of all ages. In educational applications they can be used for testing of a multiple choice type, and can be especially useful for use with children who cannot read.

In ATARI PILOT the position of the controller is "sensed" by the computer, and a USE module tells the computer what to do at that position of the controller. The Joystick Etch program included at the end of this article demonstrates the way in which the modules are used, and the "sensing" of the computer. Debbie at ATARI, who has been most helpful, sent me the information which made this program possible. Note, however, that in the computation lines, e.g. C:#X=%X, that the # and not the % must be used in order for the program to work.

We have used this program in our family not only to learn to include the joystick in our programs, but to aid in the development of graphics. Line 600 gives the location of X and Y, and if %A is added the value of THETA (angle in which the turtle is headed) is given. By using this little program, the children have been able to find locations and angles for drawing graphics in PILOT without having to resort to graph paper which they consider a laborious task.

ATARI PILOT allows the sensing of all four joysticks. The program at the end of this article is written for joystick 1. The values %J0 through %J3 are used to sense the joysticks 1 to 4 respectively; the location of the stick is indicated by the values of 1 to 10. The trigger is given the value of 1 if pressed and 0 if up (that condition was not used in this program). The instruction JUMP on condition to a specified module is then used to give the desired result.

A Paddle-Sketch program is included in the teacher's manual of the educator's package. The values for the paddle are: %P0 through %P7 for paddles 1 through 8 respectively, and the triggers have similar values: %T0 to %T7. The location desired is indicated by a value of 0 to 277. I have included a little routine to change *JOY in the JOY MATCH program to demonstrate the use of paddle 1. We found with our paddles, that the added pauses in the middle positions helped the program to sense the paddle and use the appropriate modules.

Since ATARI PILOT does not included Player Missle Graphics as a part of its program (although they can be used by the CALL command as I understand it) if one wishes to change the angle of a given graphics figure, it is necessary to change the FILL command so that the FILL is always to the right of the cursor.

I am including a program called Joystick Match as an additional example of the use of the joystick, and as an example of a more educational type use for testing. The paddle program included at the end can be used by either substituting it for *JOY or by editing the program and changing the joystick values given to the paddle values.

As a family we are in the process of developing a number of educational games using ATARI PILOT. The ones we are presently working on are for children on the preschool or early primary grades, and expect to have a disk of at least half a dozen ready by October. Anyone interested in those programs should send me a self addressed stamped envelope. My address is 84641 Hideaway Hills Road, Eugene, Oregon 97405. The games we are presently programming are used to teach number, number facts, color recognition, and upper and lower case letters. We are using graphics and the joysticks and paddles extensively in those games, and recommend their use to make programs more attractive to children. At our house we refer to it as "the joy of learning."

```
5 R:JOYSTICK ETCH  by Ruth Ellsworth
10 C:$Y=0
20 C:$X=0
100 *STICK
110 J(%J0=1):*A [JOYSTICK TOP
120 J(%J0=9):*B [JOYSTICK RIGHT TOP
130 J(%J0=8):*C [JOYSTICK RIGHT SIDE
140 J(%J0=10):*D[JOYSTICK RIGHT BOTTOM
150 J(%J0=6):*F [JOYSTICK BOTTOM
160 J(%J0=2):*E [JOYSTICK LEFT BOTTOM
170 J(%J0=4):*F [JOYSTICK LEFT SIDE
180 J(%J0=5):*H [JOYSTICK LEFT TOP
190 J(%T8=1):*M [JOYSTICK TRIGGER
191 J(%X<-79):*I
192 J(%X>79):*J
193 J(%Y<-31):*K
194 J(%Y>47):*L
240 GR:GOTO %X,%Y
250 J:*STICK
255 E:
260 *A
270 C:$X=%X
280 C:$Y=%Y+1
281 GR:DRAWTO#X,$Y
285 J:*STICK
287 E:
290 *B
300 C:$Y=%Y+1
310 C:$X=%X+1
312 GR:DRAWTO#X,$Y
315 J:*STICK
317 E:
320 *C
330 C:$Y=%Y
340 C:$X=%X+1
345 GR:DRAWTO#X,$Y
346 J:*STICK
347 E:
350 *D
360 C:$Y=%Y-1
370 C:$X=%X+1
375 GR:DRAWTO#X,$Y
376 J:*STICK
377 E:
380 *E
390 C:$Y=%Y-1
400 C:$X=%X
405 GR:DRAWTO#X,$Y
407 J:*STICK
408 E:
410 *F
420 C:$Y=%Y-1
430 C:$X=%X-1
435 GR:DRAWTO#X,$Y
437 J:*STICK
438 E:
440 *G
450 C:$Y=%Y
460 C:$X=%X-1
465 GR:DRAWTO#X,$Y
467 J:*STICK
468 E:
470 *H
480 C:$Y=%Y+1
490 C:$X=%X-1
495 GR:DRAWTO#X,$Y
497 J:*STICK
498 E:
500 *I
510 C:$X=-79
515 GR:DRAWTO-79,$Y
516 J:*STICK
517 E:
520 *J
530 C:$X=79
535 GR:DRAWTO79,$Y
536 J:*STICK
538 E:
540 *K
550 C:$Y=-31
555 GR:DRAWTO#X,-31
557 J:*STICK
558 E:
560 *L
570 C:$Y=47
575 GR:DRAWTO#X,47
577 J:*STICK
578 E:
580 *M
590 T:THE VALUE OF X AND Y
600 :ARE %X  ,%Y
601 PA:90
605 J:*STICK
607 E:
10 R:JOYSTICK MATCH
110 GR:CLEAR
120 *EX1
130 U:*PICTURE1
140 U:*PICTURE2
150 U:*PICTURE3
160 U:*PICTURE4
170 U:*PICTURE5
180 T:USE JOYSTICK TO MATCH
181 T:USE TRIGGER TO INDICATE CHOICE
190 *JOY
200 J(%T8=1):*NEXT
210 J(%J0=1):*BOX
220 J(%J0=8):*UNDERLINE
230 J(%J0=2):*LINE
240 J(%J0=4):*LITBOX
250 J:*JOY
260 E:
270 *PICTURE1
280 GR:GOTO-60,30
290 C:$A=0
300 *JUMPHERE
310 C:$A=$A+1
320 GR:4(DRAW10;TURN90)
330 GR:TURN360/8
340 J($A<8):*JUMPHERE
350 E:
360 *PICTURE2
370 GR:GOTO-21,30
380 C:$A=0
390 *TIME2
400 C:$A=$A+1
410 GR:5(DRAW15;TURN144)
420 GR:TURN72
430 J($A<5):*TIME2
440 E:
450 *PICTURE3
460 GR:GOTO21,30
470 C:$A=0
480 *TIME3
490 C:$A=$A+1
500 GR:6(DRAW 7;TURN60)
510 GR:TURN360/6
520 J($A<8):*TIME3
530 E:
540 *PICTURE4
550 GR:GOTO62,30
560 C:$A=0
570 *TIME4
580 C:$A=$A+1
590 GR:5(DRAW 7;TURN72)
600 GR:TURN360/8
610 J($A<8):*TIME4
620 E:
630 *PICTURE5
640 GR:GOTO 0,-11;TURNTO0
650 C:$A=0
660 *TIME5
670 C:$A=$A+1
680 GR:4(DRAW 10;TURN90)
690 GR:TURN360/8
700 J($A<8):*TIME5
710 E:
720 U:*JOY
```

```
730 E:
740 *BOX
750 U:*NOUL
760 U:*NOL
770 U:*NOLB
780 GR:PEN RED
790 GR:GOTO-78,13;TURNTO0
800 GR:4(DRAW 34;TURN90)
810 J:*JOY
820 E:
830 *UNDERLINE
840 U:*NOBOX
850 U:*NOL
860 U:*NOLB
870 GR:PEN RED
880 GR:GOTO-38,13;TURNTO90
890 GR:DRAW34
900 J:*JOY
910 E:
920 *LITBOX
930 U:*NOBOX
940 U:*NOUL
950 U:*NOL
960 GR:PEN RED
970 GR:GOTO50,13;TURNTO90
980 GR:DRAW 24;TURN90;DRAW4;TURN90;DRAW2
4;TURN90;FILL4
990 J:*JOY
1000 E:
1010 *LINE
1020 U:*NOBOX
1030 U:*NOUL
1040 U:*NOLB
1050 GR:PEN RED
1060 GR:GOTO 0,-11;DRAWTO21,30
1070 J:*JOY
1080 E:
1090 *NOBOX
1100 GR:PEN ERASE
1110 GR:GOTO -78,13;TURNTO0;DRAWTO-78,47
;TURN90;DRAWTO-44,47;TURN90;DRAWTO-44,13
;TURN90;DRAWTO-78,13;TURNTO0
1120 E:
1130 *NOUL
1140 GR:PEN ERASE
1150 GR:GOTO-38,13;TURNTO 0
1160 GR:DRAWTO -4,13
1170 E:
1180 *NOL
1190 GR:PEN ERASE
1200 GR:GOTO 0,-11; DRAWTO 21,30
1210 E:
1220 *NOLB
1230 GR:PEN ERASE
1240 GR:GOTO50,13;TURNTO90
1250 GR:DRAWTO74,13
1260 GR:TURN90
1270 GR:DRAW1
1280 GR:TURN90
1290 GR:DRAWTO50,12
1300 GR:TURN-90
1310 GR:DRAW1
1320 GR:TURN-90
1330 GR:DRAWTO74,11
1340 GR:TURN90
1350 GR:DRAW1
1360 GR:TURN90
1370 GR:DRAWTO50,10
1380 GR:TURN-90
1390 GR:DRAW1
1400 GR:TURN-90
1410 GR:DRAWTO74,9
1420 E:
1430 *NEXT
1440 T:TO START OVER TYPE RUN
1445 A:$X
1446 M:RUN
1447 JY:*EX1
*PADDLE (NO NUMBERS ARE INCLUDED IN THIS
 ROUTINE
J(%T0):*NEXT
J(%P0=0):*BOX
J(%P0=60):*UNDERLINE
PA:90
J(%P0=120):*LINE
PA:90
J(%P0=227):*LITBOX
J:*JOY
E:
```

DATA PERFECT REVIEW

(LJK, POB 10827, St. Louis, MO 63129, $99)

Kirt E. Stockwell

As I mantioned last month, Data Perfect is a very powerful utility. As I have become more familiar with this program, I have become convinced that it has the potential to replace a very large portion of the special-purpose software being written currently. The program responds quite fast, and the screen formatting is fairly user friendly.

There is quite good error trapping built in, but the messages or responses you get from the computer when your input is incorrect can sometimes be quite confusing. As a programmer I truly appreciate the superb quality of the programming effort required to build such a comprehensive program.

Now about the documentation....

In all fairness I must say that the documentation provided with Data Perfect is marked as a PRELIMINARY version of the manual. (The quality of the printing looks like a final copy) The only true shortcoming that I could find in Data Perfect is the manual. Considering the potential of this superb piece of software, it would be a shame if the documentation could not be re-written in order to match the quality of the program.

It might be appropriate here to mention that the vast majority of potential purchasers of Data Perfect are not programmers, nor are they adept at reading arcane documantation. To the programmer/s responsible for the actual program, Congratulations on a complex job extremely well done. To the person/s responsible for the documentation, Keep trying: the necessary information is all there, just needs some rearranging and clarification. To potential purchasers: If you need real database capabilities, you can't get more for your dollar: but you will need your patience.

RANDOM ACCESS/STRUCTURE

Kirt E. Stockwell

This month we will combine the two sets of articles, covering Random Access and Structure, that we have been presenting separately. The discussion of Random Access has reached the point where a demonstration program is in order, and a good example of some of the concepts of Structured programming in action might help to make those concepts more clear.

The example program will not be listed here in the newsletter. We will list portions that are relevant to our discussions, but will leave the bulk of the program untouched. The only way of obtaining the complete program currently is to send for the special Utility disk that we are promoting to raise funds for our dedicated Bulletin Board System. If you get this program and list it out, preferrably on paper, you will be able to see the Structure quite clearly.

The program requires almost 10K of free ram, plus another 5K for working storage. This might seem like a heavy RAM requirement, until you realize that this will allow you to store 150 records (each 105 characters long). If you have adequate memory to provide 15K of free Ram after the program is loaded, you will be able to store 500 records on disk, and search the entire file very rapidly, as the keyfile is resident in RAM. This provides extremely fast searching and retrieval of desired records, and all records can be edited individually and re-recorded without having to wade through the whole database.

A portion of the program will be presented below. This is actually 2 separate control modules found within the body of the program. I am sure that there are programmers out there who could write a complex program such as this without using structured design, but not many of them.

The 2 control modules shown will provide an example of how control can be directed between discreet modules in a program and how they can be linked together. You will notice that there are separate control modules that are subordinate to the master control module. By dividing the control process up in this manner, it becomes possible to run and/or test discreet components of the program even if other components are not yet completed. This also facilitates changes when they are needed. (you wouldn't believe how many changes are needed in a program like this one)

Look over the listings shown below, and you will see how the master control module and a subordinate control module work together. Next month, some of the reasons why this fool program is so long.....

```
10 REM PHONEBUK:RANDOM ACCESS SAMPLE
20 REM By Kirt E. Stockwell,President
30 REM Eugene A.C.E. (ACE International)
100 COVER=19000:SDIMWIT=6000
110 GOSUB COVER
120 GOSUB SDIMWIT
130 GOSUB FINDER
140 GOSUB MENUTOP
160 FOR LOOPS=1 TO MAXRECS
170 GOSUB GITPICK
180 ON S GOSUB ENTRY,EDIT,SEARCH,QUIT
190 NEXT LOOPS
195 END
```

4000
~~4150~~
415F

351
16
8
40

256
60
15
1

BALLOONS

by Stan Ockers, Lockport, IL

This month we look at an expansion of the PM graphics routine introduced in June's 'Bankshot' program. I changed the routine slightly so that a vertical position change wouldn't be necessary to change the images. This was done using old and new image numbers stored in page six and required adjustment of the page six table. Table 1 is a revised listing of page six locations used.

You will also notice a number of flag locations in table 1. Additional machine language routines have been added prior to the main X-Y movement routine. Communication between these routines and Basic is done by setting and resetting flags. In general, the flag for a given routine will be set to one by Basic or another routine. The flag being one is an indication that the routine is active. Upon completion of certain actions, the flag will be reset to zero.

In the game 'Balloons' someone is tossing water filled balloons across the screen. Your job is to fire darts trying to bust the ballons. Don't hit the birds though, you'll lose points rather than gain them. Allowances must be made for the wind which changes speed and direction every round. At the beginning of the game you are offered a menu you can use to change playing parameters of the game. Try the default parameters first to get used to the game and then try varying parameters.

The whole game could have been written using just the main X-Y movement routine (data in lines 270-302) as 'Bankshot' was, but because of the timing between Basic and the Vertical Blank Interrupt, motion would not be smooth. For smooth motion seperate routines have been put in the VBI. The three balloons (3 players) have their horizontal position updated by a routine using data from lines 262 and 263. Speed is determined by bytes in page 6 (1572-1574) with negative numbers indicating motion to the left. When players reach specified limits their horizontal positions are set to zero to remove them from the screen.

Another routine (data from 264 to 266) handles dart movement. The number of vertical moves before a horizontal one is specified by WIND (1576) while direction is indicated by DIR (1577) and speed by SPED3 (1575). When a vertical limit is reached the dart is replace at the starting position.

A check is made for collisions and the player hit is removed to the left side of the screen (routine data is in line 267 and part of 268). FLAG4 (1578) is set indicating a collision has taken place. Basic picks this up and sets FLAG5 (1579) after producing the sounds and images of the collision. FLAG5 starts the fall routine (remainder of 268 and 269) which drops the image to the bottom of the screen and resets flags 4 and 5, putting a dart at the beginning position.

One last job for the VBI routine .... In line 261 is a routine that puts the horizontal position of the players in sound frequency registers. It also puts the vertical position of the dart into a sound register if the dart movement routine is active. These actions help considerably in generating smoothly varying pitches which would be impossible using Basic because pokes are not fast enough.

Table 1

0600-0609 (1536-1545) VBI insertion routine
060A (1546) Temp variable
060B (1547) PM area Hi byte
060C-060F (1548-1551) Old image numbers
0610-0613 (1552-1555) New image numbers
0614-0617 (1556-1559) Horizontal positions
0618-061B (1560-1563) Old vertical positions
061C-061F (1564-1567) New vertical positions
0620-0623 (1568-1571) Flags for players
0624-0627 (1572-1575) Speedof players
0628 (1576) WIND
0629 (1577) DIR
062A (1578) FLAG4
062B (1579) FLAG5
062C-062F Reserved for future use
0630 and up (1584) Image pointers Lo & Hi bytes

```
10 REM xxxxxxxxxxxxxxxxxxxx
20 REM xx   BALLOONS    xx
30 REM xxSTAN OCKERS 6/2xx
35 REM xx   AUG 1982    xx
40 REM xxxxxxxxxxxxxxxxxxxx
90 ? "JUST A SEC..."
100 PMHI=1547:IMAGE0=1552:IMAGE1=1553:IMAGE2=
1554:IMAGE3=1555:HPOS3=1559:VPOS3=1567
110 FLAG0=1568:FLAG1=1569:FLAG2=1570:FLAG3=15
71:FLAG4=1578:FLAG5=1579:WIND=1576:DIR=1577
120 SPEED0=1572:SPEED1=1573:SPEED2=1574:SPEED
3=1575:IMGPNT=1584:SIZEP3=53259:P3PL=53263:AU
DC1=53761
130 AUDC2=53763:AUDC3=53765:AUDC4=53767:RAMTO
P=106:PMBASE=54279:SDMCTL=559:GRACTL=53277
140 PCOLR0=704:PCOLR1=705:PCOLR2=706:PCOLR3=7
07
150 NR=5:ROUND=1:DIF=4:DLY=10:DSPD=10:WAIT=20
:AMMO=15:NP=1:BPR=0.4
200 OPEN #1,4,0,"K"
250 REM x VBI ROUTINE UPDATED BY POKES x
260 DIM VB$(348):FOR I=1 TO 348:READ A:VB$(I,
I)=CHR$(A):NEXT I
261 DATA 173,20,6,141,0,210,173,21,6,141,2,21
0,173,22,6,141,4,210,173,35,6,240,6,173,31,6,
141,6,210
262 DATA 162,2,189,32,6,240,46,188,36,6,240,4
1,48,15,254,20,6,136,208,250,189,20,6,201,200
,144,26,176,13
263 DATA 222,20,6,200,208,250,189,20,6,201,32
,176,8,169,0,157,20,6,157,36,6,157,32,6,202,1
6,202
264 DATA 173,35,6,240,60,174,39,6,173,10,6,20
8,19,173,40,6,141,10,6,173,41,6,208,5,206,23,
6
265 DATA 16,3,238,23,6,206,31,6,206,10,6,173,
31,6,201,32,176,17,169,120,141,23,6,169,224,1
41,31,6
266 DATA 169,0,141,35,6,240,3,202,16,199
267 DATA 173,42,6,208,31,173,15,208,240,26,16
2,0,74,176,5,232,224,4,208,248,169,0,141,35,6
,157,32,6,157,20,6
268 DATA 169,1,141,42,6,173,43,6,240,36,238,3
1,6,173,31,6,201,224,144,26,141,30,208,169,12
0
269 DATA 141,23,6,169,224,141,31,6,169,4,141,
19,6,169,0,141,42,6,141,43,6
270 DATA 24,173,11,6,105,4,133,204,162,0,134,
207,160,8,132,203,189,20,6,157,0,208,189,12,6
,221,16,6,208,8,189,28,6
280 DATA 221,24,6,240,69,189,16,6,157,12,6,18
9,28,6,157,24,6,165,203,221,28,6,240,10,169,0
,145,203,230,203,240,42
290 DATA 208,239,189,16,6,170,189,48,6,133,20
5,189,49,6,133,206,177,205,240,14,145,203,230
,205,208,2
300 DATA 230,206,230,203,240,10,208,238,169,0
,145,203,230,203,208,250,230,204,166,207,232,
134,207,224,4,144,154
302 DATA 76,98,228
350 REM x PAGE 6 - INSERT VBI ROUTINE x
360 FOR I=1536 TO 1545:READ A:POKE I,A:NEXT I
370 DATA 104,160,0,162,0,169,7,76,92,228
380 A=ADR(VB$):B=INT(A/256):C=A-256xB:POKE 15
38,C:POKE 1540,B
400 GRAPHICS 17:DIM N$(3)
430 GOSUB 1800:DIM T(NP)
500 REM x PLAYER MISSILE SETUP x
510 A=PEEK(RAMTOP)-16:POKE PMBASE,A:POKE PMHI
,A
560 REM xx IMAGE 14 BIRD FALL xx
562 DIM BFALL$(11):FOR I=1 TO 11:READ A:BFALL
$(I,I)=CHR$(A):NEXT I:BFL=ADR(BFALL$):POKE IM
GPNT+15,INT(BFL/256)
564 POKE IMGPNT+14,BFL-256xPEEK(IMGPNT+15)
566 DATA 68,68,68,40,40,56,56,16,16,16,0
570 REM xx IMAGE 12 BIRD SPLAT xx
572 DIM BOOM3$(11):FOR I=1 TO 11:READ A:BOOM3
$(I,I)=CHR$(A):NEXT I:BM3=ADR(BOOM3$):POKE IM
GPNT+13,INT(BM3/256)
574 POKE IMGPNT+12,BM3-256xPEEK(IMGPNT+13)
576 DATA 16,40,146,214,124,124,56,56,16,40,0
580 REM xx IMAGE 10 BALLOON BURST xx
582 DIM BOOM2$(14):FOR I=1 TO 14:READ A:BOOM2
$(I,I)=CHR$(A):NEXT I:BM2=ADR(BOOM2$):POKE IM
GPNT+11,INT(BM2/256)
584 POKE IMGPNT+10,BM2-256xPEEK(IMGPNT+11)
586 DATA 8,40,52,84,74,42,33,17,84,76,42,50,1
6,0
590 REM xx IMAGE 8 BIRD xx
592 DIM BIRD$(9):FOR I=1 TO 9:READ A:BIRD$(I,
I)=CHR$(A):NEXT I:BRD=ADR(BIRD$):POKE IMGPNT+
9,INT(BRD/256)
594 POKE IMGPNT+8,BRD-256xPEEK(IMGPNT+9)
596 DATA 192,96,112,58,63,126,60,24,0
600 REM xx IMAGE 6 RAIN xx
610 DIM RAIN$(10):FOR I=1 TO 10:READ A:RAIN$(
I,I)=CHR$(A):NEXT I:RAI=ADR(RAIN$):POKE IMGPN
T+7,INT(RAI/256)
612 POKE IMGPNT+6,RAI-256xPEEK(IMGPNT+7)
614 DATA 32,4,8,66,32,133,32,8,68,0
620 REM xx IMAGE 4 DART xx
624 DIM DART$(9):FOR I=1 TO 9:READ A:DART$(I,
I)=CHR$(A):NEXT I:DAR=ADR(DART$):POKE IMGPNT+
5,INT(DAR/256)
625 POKE IMGPNT+4,DAR-256xPEEK(IMGPNT+5)
626 DATA 16,16,16,16,16,56,56,40,0
628 REM xx IMAGE 2 xx
630 DIM BALL$(11):FOR I=1 TO 11:READ A:BALL$(
I,I)=CHR$(A):NEXT I:BAL=ADR(BALL$):POKE IMGPN
T+3,INT(BAL/256)
632 POKE IMGPNT+2,BAL-256xPEEK(IMGPNT+3)
640 DATA 36,24,52,122,126,126,126,126,60,24,0
642 REM xx IMAGE 0 xx
```

BALLOONS

```
645 DIM Z$(1):Z$=CHR$(0):ZERO=ADR(Z$):POKE IM
GPNT+1,INT(ZERO/256):POKE IMGPNT,ZERO-256*PEE
K(IMGPNT+1)
649 REM xx INIT. IMAGES, HOR. & VERT. POS. xx
650 FOR I=1552 TO 1567:READ A:POKE I,A:NEXT I
660 DATA 2,2,2,4,0,0,0,120,0,0,0,0,50,90,130,
230
669 REM xx PM GRAPHICS - COLORS xx
670 POKE SDMCTL,62:POKE GRACTL,3
695 POKE PCOLR0,38:POKE PCOLR1,56:POKE PCOLR2
,70:POKE PCOLR3,86
700 A=USR(1536)
720 RESTORE 725:DIM MPH$(5),SPD$(8):FOR I=1 T
O 5:READ A:MPH$(I,I)=CHR$(A):NEXT I
722 FOR I=1 TO 8:READ A:SPD$(I,I)=CHR$(A):NEX
T I:POKE FLAG3,0:POKE FLAG4,0:POKE FLAG5,0
725 DATA 3,6,9,14,20,254,1,253,2,251,3,250,4
726 FOR I=1 TO NP:T(I)=0:NEXT I:PLYR=0:GOSUB
1400
728 POKE SPEED3,DSPD:SHOTS=AMMO:GOTO 732
729 REM xx MAIN LOOP STARTS HERE xx
730 GOSUB 1600:POKE 77,0:SOUND 3,0,0,0:GOSUB
1700:SHOTS=AMMO
732 ? #6;CHR$(125):POSITION 1,0:? #6;"XXXXXXX
XXXXXXXXXX ":REM X'S IN INVERSE
734 SX=3:SY=19:FOR I=1 TO SHOTS:GOSUB 1100:?
#6;"x";:NEXT I:SX=3:SY=19
740 R0=RND(0)*DLY:R1=RND(0)*DLY:R2=RND(0)*DLY
742 T(PLYR)=T(PLYR)+TOT:TOT=0
756 POSITION 4,18:? #6;"shots":POSITION 13,18
:? #6;"wind":POSITION 12,22:? #6;"score"
762 PLYR=PLYR+1:IF PLYR>NP THEN PLYR=1:ROUND=
ROUND+1:GOSUB 1400:IF ROUND>NR THEN 1000
764 POSITION 12,19:IF R8=0 THEN FOR I=1 TO 6-
R9:? #6;"<";:NEXT I
766 IF R8=1 THEN FOR I=1 TO 6-R9:? #6;">";:NE
XT I
780 POSITION 2,5:? #6;"PULL STICK TO GO"
782 POSITION 12,21:? #6;T(PLYR)
784 FOR J=1 TO 30:NEXT J:POSITION 3,23:? #6;"
                "
786 FOR J=1 TO 30:NEXT J:POSITION 11,23:? #6;
"plyr #";PLYR:POSITION 3,23:? #6;"rnd #";ROUN
D
788 IF STICK(0)<>13 THEN 784
790 POSITION 2,5:? #6;"                "
800 IF SHOTS=0 THEN 830
810 IF STRIG(0)=0 AND PEEK(FLAG3)=0 AND PEEK(
FLAG4)=0 THEN POKE FLAG3,1:SHOTS=SHOTS-1:GOSU
B 1100:? #6;" ";
820 IF PEEK(FLAG3)=1 THEN POKE AUDC4,168
830 IF PEEK(FLAG3)=0 THEN POKE AUDC4,0
835 R=INT(RND(0)*DIF)+1:SP=ASC(SPD$(R))
840 IF PEEK(FLAG0)=0 THEN R0=R0-1:POKE AUDC1,
0
842 IF R0<0 THEN R0=RND(0)*DLY:POKE FLAG0,1:P
OKE SPEED0,SP:POKE IMAGE0,2:E0=0:IF RND(0)<BP
R THEN GOSUB 1300
844 IF PEEK(FLAG0)=1 THEN POKE AUDC1,162
850 IF PEEK(FLAG1)=0 THEN R1=R1-1:POKE AUDC2,
0
852 IF R1<0 THEN R1=RND(0)*DLY:POKE FLAG1,1:P
OKE SPEED1,SP:POKE IMAGE1,2:E1=0:IF RND(0)<BP
R THEN GOSUB 1310
854 IF PEEK(FLAG1)=1 THEN POKE AUDC2,162
860 IF PEEK(FLAG2)=0 THEN R2=R2-1:POKE AUDC3,
0
862 IF R2<0 THEN R2=RND(0)*DLY:POKE FLAG2,1:P
OKE SPEED2,SP:POKE IMAGE2,2:E2=0:IF RND(0)<BP
R THEN GOSUB 1320
864 IF PEEK(FLAG2)=1 THEN POKE AUDC3,162
870 IF PEEK(FLAG4)=1 AND PEEK(FLAG5)=0 THEN G
OSUB 1500:POKE IMAGE3,6:POKE FLAG5,1
880 IF PEEK(FLAG5)=1 THEN SOUND 3,PEEK(VPOS3)
,10,8:IF FLAG6=1 THEN POKE IMAGE3,14
890 IF PEEK(FLAG5)=0 THEN POKE IMAGE3,4
900 IF SHOTS=0 THEN PTIM=0:TIME=WAIT:GOTO 730
910 TIME=TIME-1:IF TIME<0 THEN TIME=WAIT:GOSU
B 1200:IF PTIM>18 THEN PTIM=0:GOTO 730
920 GOTO 800
999 REM xx FINAL SCORES xx
1000 ? #6;CHR$(125):POSITION 3,3:? #6;"final
scores":X=3:Y=5:FOR I=1 TO NP:POSITION X,Y
1005 ? #6;"PLYR # ";I;"=";T(I):Y=Y+1:NEXT I
1010 GOTO 1010
1099 REM xx POS. FOR PRINTING 'x' xx
1100 SX=SX+1:IF SX=9 THEN SX=4:SY=SY+1
1110 POSITION SX,SY:RETURN
1199 REM xx POS. FOR PRINTING 'x' xx
1200 PTIM=PTIM+1:POSITION PTIM,0:? #6;"x";:RE
TURN
1299 REM xx CHANGE TO A BIRD SUBR. xx
1300 POKE IMAGE0,8:POKE SPEED0,1:E0=1:RETURN
1310 POKE IMAGE1,8:POKE SPEED1,1:E1=1:RETURN
1320 POKE IMAGE2,8:POKE SPEED2,1:E2=1:RETURN
1399 REM xx WIND SPEED AND DIR SUBR. xx
1400 R9=INT(RND(0)*5+1):MPH=ASC(MPH$(R9)):POK
E WIND,MPH:R8=INT(RND(0)*2):POKE DIR,R8:RETUR
N
1499 REM xx HIT BIRD OR BALLOON SUBR. xx
1500 PHIT=PEEK(P3PL):IF (E0=0 AND E1=0) AND E
2=0 THEN GOTO 1508
1502 IF (PHIT=1 AND E0=0) OR (PHIT=2 AND E1=0
) OR (PHIT=4 AND E2=0) THEN 1508
1504 FOR I=1 TO 3:POKE IMAGE3,12:SOUND 3,10*I
+10,10,10:FOR J=1 TO 3:NEXT J:POKE IMAGE3,0
1506 FOR J=1 TO 3:NEXT J:NEXT I:FLAG6=1:GOTO
1514
1508 H3=PEEK(HPOS3):RESTORE 1530:FOR I=1 TO 3
:READ M,X:POKE HPOS3,H3-X:POKE SIZEP3,M:POKE
IMAGE3,10
1510 SOUND 3,100-10*I,12,10:POKE 1555,0:FOR J
=1 TO 30:NEXT J:NEXT I:POKE 53259,0:FLAG6=0:P
OKE 1559,H3
1514 IF PHIT=1 AND E0=1 THEN E0=0:TOT=TOT-60
1516 IF PHIT=2 AND E1=1 THEN E1=0:TOT=TOT-40
1518 IF PHIT=4 AND E2=1 THEN E2=0:TOT=TOT-20
1520 IF PHIT=1 AND E0=0 THEN TOT=TOT+30
1522 IF PHIT=2 AND E1=0 THEN TOT=TOT+20
1524 IF PHIT=4 AND E2=0 THEN TOT=TOT+10
1526 POSITION 12,21:? #6;T(PLYR)+TOT;"  "
1528 RETURN
1530 DATA 0,0,1,8,3,16
1599 REM xx DART TO INIT. POS. xx
1600 POKE FLAG3,0:POKE IMAGE3,4:POKE HPOS3,12
0:POKE VPOS3,230:POKE FLAG6,0:RETURN
1699 REM xx ROUND OVER SUBR. xx
1700 FOR I=1 TO 10:FOR J=1 TO 30:NEXT J:POSIT
ION 5,6:IF SHOTS=0 THEN ? #6;"out of shots":G
OTO 1720
1710 ? #6;"out of time  "
1720 POSITION 12,21:? #6;T(PLYR)+TOT:FOR J=1
TO 30:NEXT J:POSITION 5,6:? #6;"
"
1730 POSITION 12,21:? #6;"    ":NEXT I:RETUR
N
1799 REM xx ENTER PARAMETERS SUBR. xx
1800 ? #6;CHR$(125):POSITION 3,3:? #6;"PLAYIN
G VALUES":POSITION 3,5:? #6;"enter one #"
1810 POSITION 3,7:? #6;"(1)# SHOTS =";AMMO:PO
SITION 3,8:? #6;"(2)DART SP.=";DSPD
1820 POSITION 3,9:? #6;"(3)BAL. SP.=";DIF:POS
ITION 3,10:? #6;"(4)BAL. DLY=";DLY
1830 POSITION 3,11:? #6;"(5)RND TIME=";WAIT:P
OSITION 3,12:? #6;"(6)# ROUNDS=";NR
1840 POSITION 3,13:? #6;"(7)# PLYRS =";NP
1842 POSITION 3,14:? #6;"(8)BIRDS   =";BPR
1845 POSITION 3,15:? #6;"(9) NO CHANGES"
1850 GET #1,A:IF A<49 OR A>57 THEN 1850
1855 TRAP 1800
1860 ? #6;CHR$(125):POSITION 3,6:K=A-48:GOSUB
 1900+K*10
1870 IF K=9 THEN RETURN
1880 GOTO 1800
1910 ? #6;"# SHOTS = ";AMMO:POSITION 6,10:? #
6;"(1 TO 20)":GOSUB 1991:AMMO=A:RETURN
1920 ? #6;"DART SPEED =";DSPD:POSITION 3,10:?
 #6;"1(SLOW)-20(FAST)":GOSUB 1991:DSPD=A:RETU
RN
1930 ? #6;"BAL. SPEED =";DIF:POSITION 3,10:?
#6;"1(SLOW)-8(FAST)":GOSUB 1991:DIF=A:RETURN
1940 ? #6;"BAL. DELAY =";DLY:POSITION 2,10:?
#6;"1(SHORT)-50(LONG)":GOSUB 1991:DLY=A:RETUR
N
1950 ? #6;"RND TIME =";WAIT:POSITION 3,10:? #
6;"5(FAST)-30(SLOW)":GOSUB 1991:WAIT=A:RETURN

1960 ? #6;"# ROUNDS =";NR:POSITION 4,10:? #6;
"(1-WHATEVER)":GOSUB 1991:NR=A:RETURN
1970 ? #6;"# PLAYERS = ";NP:POSITION 5,10:? #
6;"(1 OR MORE)":GOSUB 1991:NP=A:RETURN
1980 ? #6;"BIRD PROB. = ";BPR:POSITION 5,10:?
 #6;"(0.1 TO 0.9)":GOSUB 1991:BPR=A:RETURN
1990 RETURN
1991 N$="":POSITION 3,8:? #6;"INPUT NEW VALUE
 ";:POSITION 8,12
1992 GET #1,A:? #6;CHR$(A);:IF A=155 THEN A=V
AL(N$):RETURN
1994 N$(LEN(N$)+1)=CHR$(A):GOTO 1992
```

Note: In the "Bankshot" VBI routine in the June issue of ACE, the branch (BNE) at the end doesn't always work right, especially if more than two players are used. Try changing the BNE to BCC. In hex, the D0 should be 90 or in decimal, the 208 (next to the last byte) in line 300 should be changed to 144.

-Stan Ockers

The following source listings are to help with understanding of the machine language sub-routes on the BASIC program and are not typed in by you. The DATA statements in the BASIC porgram are the decimal equivalents of the following (ed.)

Sound

```
          10 ; ROUTINE TO FILL SOU. FREQ. REG.
D200      20 AUDF1   =   $D200
D202      30 AUDF2   =   $D202
D204      40 AUDF3   =   $D204
D206      50 AUDF4   =   $D206
0614      60 HPOS0   =   $0614
0615      70 HPOS1   =   $0615
0616      80 HPOS2   =   $0616
0623      90 FLAG3   =   $0623
061F      0100 VPOS3 =   $061F
0000      0110       *=  $0000
0000 AD1406 0120     LDA HPOS0    Transfer horiz.
0003 8D00D2 0130     STA AUDF1    pos. to freq.
0006 AD1506 0140     LDA HPOS1    registers
0009 8D02D2 0150     STA AUDF2
000C AD1606 0160     LDA HPOS2
000F 8D04D2 0170     STA AUDF3
0012 AD2306 0180     LDA FLAG3    Is dart active?
0015 F006   0190     BEQ OUT      No
0017 AD1F06 0200     LDA VPOS3    Transfer vert.
001A 8D06D2 0210     STA AUDF4    to freq.
001D EA     0220 OUT NOP
```

```
=D200 AUDF1      =D202 AUDF2      =D204 AUDF3      =D206 AUDF4
=0614 HPOS0      =0615 HPOS1      =0616 HPOS2      =0623 FLAG3
=061F VPOS3       001D OUT
```

Hor3

```
          10 ; ROUTINE TO MOVE 3 PLAYERS HORIZ.
0624      20 SPEED   =   $0624
0614      30 HPOS    =   $0614
0620      40 FLAG    =   $0620
00C8      50 RTLIM   =   $C8
0020      60 LTLIM   =   $20
0000      70         *=  $0000
0000 A202   80       LDX #$02     Three players
0002 BD2006 90 MOVE  LDA FLAG,X   Ck for movement
0005 F02E   0100     BEQ NXT      No move
0007 BC2406 0110     LDY SPEED,X  Get speed in index
000A F029   0120     BEQ NXT      If zero, no movement
000C 300F   0130     BMI REV      If neg. go backwards
000E FE1406 0140 FOR INC HPOS,X   One step forward
0011 88     0150     DEY          More?
0012 D0FA   0160     BNE FOR      Yes
0014 BD1406 0170     LDA HPOS,X   Get horiz. pos.
0017 C9C8   0180     CMP #RTLIM   Too far right?
0019 901A   0190     BCC NXT      No
001B B00D   0200     BCS STOP     Skip over
001D DE1406 0210 REV DEC HPOS,X   Back one step
0020 C8     0220     INY          More?
0021 D0FA   0230     BNE REV      Yes
0023 BD1406 0240     LDA HPOS,X   Get horiz. pos.
0026 C920   0250     CMP #LTLIM   Too far left?
0028 B00B   0260     BCS NXT      No
002A A900   0270 STOP LDA #$00
002C 9D1406 0280     STA HPOS,X   Put player off screen
002F 9D2406 0290     STA SPEED,X  and stop movement
0032 9D2006 0300     STA FLAG,X
0035 CA     0310 NXT DEX          More players?
0036 10CA   0320     BPL MOVE     Yes
```

```
=0624 SPEED      =0614 HPOS       =0620 FLAG       =00C8 RTLIM
=0020 LTLIM       0002 MOVE        0035 NXT         001D REV
000E FOR         002A STOP
```

Wind

```
          10 ; ROUTINE TO MOVE DART
D00F      20 P3PL    =   $D00F
0020      30 VLIM    =   $20
0623      40 FLAG3   =   $0623
0628      50 WIND    =   $0628
0629      60 DIR     =   $0629
0627      70 SPEED3  =   $0627
060A      80 TEMP1   =   $060A
0078      90 STARTX  =   $78
00E0      0100 STARTY =  $E0
0617      0110 HPOS3 =   $0617
061F      0120 NEWV3 =   $061F
0000      0130       *=  $0000
0000 AD2306 0140     LDA FLAG3    Test for dart move
0003 F03C   0150     BEQ HIT      If pos. no move
0005 AE2706 0160     LDX SPEED3   How many times thru?
0008 AD0A06 0170 MOVE2 LDA TEMP1  Get counter
000B D013   0180     BNE VMOV     If not zero, only vert.
000D AD2806 0190     LDA WIND     How many vert before horiz.?
0010 8D0A06 0200     STA TEMP1    Reset counter
0013 AD2906 0210     LDA DIR      Which way to move horiz.?
0016 D005   0220     BNE RIGHT    Go right
0018 CE1706 0230     DEC HPOS3    Else go left once
001B 1003   0240     BPL VMOV
001D EE1706 0250 RIGHT INC HPOS3  One space right
0020 CE1F06 0260 VMOV DEC NEWV3   One space up
0023 CE0A06 0270     DEC TEMP1    count for horiz. move
0026 AD1F06 0280     LDA NEWV3    Have we reached limit?
0029 C920   0290     CMP #VLIM
002B B011   0300     BCS AGAIN    No, skip
002D A978   0310     LDA #STARTX  Yes, put back at start
002F 8D1706 0320     STA HPOS3
0032 A9E0   0330     LDA #STARTY
0034 8D1F06 0340     STA NEWV3
0037 A900   0350     LDA #$00     Set flag so routine stops
0039 8D2306 0360     STA FLAG3
003C F003   0370     BEQ HIT      No more movement
003E CA     0380 AGAIN DEX        Go thru again?
003F 10C7   0390     BPL MOVE2    Yes
0041 EA     0400 HIT NOP
```

```
=D00F P3PL       =0020 VLIM       =0623 FLAG3      =0628 WIND
=0629 DIR        =0627 SPEED3     =060A TEMP1      =0078 STARTX
=00E0 STARTY      =0617 HPOS3      =061F NEWV3       0041 HIT
0008 MOVE2       0020 VMOV         001D RIGHT       003E AGAIN
```

Hit

```
          10 ; ROUTINE TO HANDLE COLLISIONS
D00F      20 P3PL    =   $D00F
0620      30 FLAG0   =   $0620
0623      40 FLAG3   =   $0623
0614      50 HPOS    =   $0614
062A      60 FLAG4   =   $062A
062C      70 SCORE   =   $062C
0000      80         *=  $0000
0000 AD2A06 90       LDA FLAG4    Is hit already active
0003 D01F   0100     BNE OUT1
0005 AD0FD0 0110     LDA P3PL     Any collisions?
0008 F01A   0120     BEQ OUT1     No
000A A200   0130     LDX #$00     Find out which
000C 4A     0140 SHIFT LSR A      player
000D B005   0150     BCS FOUND    This is the one
000F E8     0160     INX          Next player
0010 E004   0170     CPX #$04     Last player?
0012 D0F8   0180     BNE SHIFT    No, try another
0014 A900   0190 FOUND LDA #00
0016 8D2306 0200     STA FLAG3    Stop dart
0019 9D2006 0210     STA FLAG0,X  Stop player and
001C 9D1406 0220     STA HPOS,X   remove from screen
001F A901   0230     LDA #$01     Signal that hit
0021 8D2A06 0240     STA FLAG4    has taken place
0024 EA     0250 OUT1 NOP
```

```
=D00F P3PL       =0620 FLAG0      =0623 FLAG3      =0614 HPOS
=062A FLAG4      =062C SCORE       0024 OUT1       000C SHIFT
0014 FOUND
```

Fall

```
          10 ; ROUTINE TO CAUSE FALL
D01E      20 HITCLR  =   $D01E
062A      30 FLAG4   =   $062A
062B      40 FLAG5   =   $062B
0617      50 HPOS3   =   $0617
061F      60 NEWV3   =   $061F
00E0      70 BTLIM   =   $E0
0078      80 STARTX  =   $78
00E0      90 STARTY  =   $E0
0004      0100 DART  =   $04
0613      0110 IMAGE3 =  $0613
0000      0120       *=  $0000
0000 AD2B06 0130     LDA FLAG5    Fall routine active?
0003 F024   0140     BEQ OUT2     No
0005 EE1F06 0150     INC NEWV3    Go down one space
```

```
0008 AD1F06 0160        LDA  NEWV3    Have we reached
000B C9E0   0170        CMP  #BTLIM   bottom?
000D 901A   0180        BCC  OUT2     Not yet
000F 8D1ED0 0190        STA  HITCLR   Clear coll. reg.
0012 A978   0200        LDA  #STARTX  and put back to origin
0014 8D1706 0210        STA  HPOS3
0017 A9E0   0220        LDA  #STARTY
0019 8D1F06 0230        STA  NEWV3
001C A904   0240        LDA  #DART    Change image to
001E 8D1306 0250        STA  IMAGE3   dart
0021 A900   0260        LDA  #$00     Clear flags
0023 8D2A06 0270        STA  FLAG4
0026 8D2B06 0280        STA  FLAG5
0029 EA     0290 OUT2   NOP

=D01E HITCLR      =062A FLAG4       =062B FLAG5       =0617 HPOS3
=061F NEWV3       =00E0 BTLIM       =0078 STARTX      =00E0 STARTY
=0004 DART        =0613 IMAGE3       0029 OUT2
```

VBI

```
                10 ; ROUTINE FOR X & Y POS. WITH POKES
0614            20 HPOS   =  $0614
0618            30 OLDV   =  $0618
061C            40 NEWV   =  $061C
00CB            50 APTR   =  $00CB
00CD            60 IPTR   =  $00CD
060B            70 PMPG   =  $060B
060C            80 IMGO   =  $060C
0610            90 IMGN   =  $0610
0630           100 IMGP   =  $0630
00CF           110 STOR   =  $00CF
D000           120 PLYRH  =  $D000
0000           130        x=  $0000
0000 18     0140 VBI    CLC
0001 AD0B06 0150        LDA  PMPG     ADD TO PM HI ...
0004 6904   0160        ADC  #$04     FOUR PAGES TO GET ...
0006 85CC   0170        STA  APTR+1   PLYR AREA PTR HI
0008 A200   0180        LDX  #$00     X IS PLYR COUNTER
000A 86CF   0190        STX  STOR
000C A000   0200        LDY  #$00     Y USED IN INDIR OPER.
000E 84CB   0210        STY  APTR
0010 BD1406 0220 HORIZ  LDA  HPOS,X   UPDATE HOR. POS.
0013 9D00D0 0230        STA  PLYRH,X
0016 BD0C06 0240        LDA  IMGO,X   HAS IMAGE CHANGED?
0019 DD1006 0250        CMP  IMGN,X
001C D008   0260        BNE  UPDATE   YES
001E BD1C06 0270        LDA  NEWV,X   CHECK IF VERT. POS.
0021 DD1806 0280        CMP  OLDV,X   CHANGED
0024 F045   0290        BEQ  NXTPLYR  SKIP IF NOT
0026 BD1006 0300 UPDATE LDA  IMGN,X   UPDATE OLD ..
0029 9D0C06 0310        STA  IMGO,X   IMAGE
002C BD1C06 0320        LDA  NEWV,X   UPDATE OLD VERT.
002F 9D1806 0330        STA  OLDV,X   POSITION
0032 A5CB   0340 ZERO1  LDA  APTR     CK. IF REACHED ..
0034 DD1C06 0350        CMP  NEWV,X   VERTICAL POS. YET
0037 F00A   0360        BEQ  LDPLYR   IF SO, START LOAD
0039 A900   0370        LDA  #$00     ELSE FILL WITH ...
003B 91CB   0380        STA  (APTR),Y ZEROS...
003D E6CB   0390        INC  APTR     UNTIL YOU DO
003F F02A   0400        BEQ  NXTPLYR  (OUT OF PLYR AREA)
0041 D0EF   0410        BNE  ZERO1
0043 BD1006 0420 LDPLYR LDA  IMGN,X   USE IMAGE NUMBER ...
0046 AA     0430        TAX           AS INDEX AND ...
0047 BD3006 0440        LDA  IMGP,X   SET UP INDIR.
004A 85CD   0450        STA  IPTR     POINTER
004C BD3106 0460        LDA  IMGP+1,X
004F 85CE   0470        STA  IPTR+1
0051 B1CD   0480 PLBYTE LDA  (IPTR),Y GET IMAGE...
0053 F00E   0490        BEQ  CLREST   IF A ZERO, FINISHED
0055 91CB   0500        STA  (APTR),Y PUT IN PLYR AREA
0057 E6CD   0510        INC  IPTR     NEXT IMAGE BYTE
0059 D002   0520        BNE  UPONE    NO PAGE CROSSING
005B E6CE   0530        INC  IPTR+1   PAGE CROSSING
005D E6CB   0540 UPONE  INC  APTR     NEXT AREA BYTE
005F F00A   0550        BEQ  NXTPLYR  REACHED END OF AREA
0061 D0EE   0560        BNE  PLBYTE   NEXT IMAGE BYTE
0063 A900   0570 CLREST LDA  #$00     ZERO OUT REST ..
0065 91CB   0580 ZERO2  STA  (APTR),Y OF PLYR AREA
0067 E6CB   0590        INC  APTR
0069 D0FA   0600        BNE  ZERO2
006B E6CC   0610 NXTPLYR INC APTR+1   NEXT AREA
006D A6CF   0620        LDX  STOR     RECOVER PLYR INDEX
006F E8     0630        INX           NEXT PLYR
0070 86CF   0640        STX  STOR     SAVE INDEX
0072 E004   0650        CPX  #$04     LAST PLAYER?
0074 909A   0660        BCC  HORIZ    NO, GET ANOTHER
0076 4C62E4 0670        JMP  $E462    BACK TO ATARI'S VBI
0079        0680        .END

=0614 HPOS        =0618 OLDV        =061C NEWV        =00CB APTR
=00CD IPTR        =060B PMPG        =060C IMGO        =0610 IMGN
=0630 IMGP        =00CF STOR        =D000 PLYRH        0000 VBI
0010 HORIZ        0026 UPDATE       006B NXTPLYR      0032 ZERO1
0043 LDPLYR       0051 PLBYTE       0063 CLREST       005D UPONE
0065 ZERO2
```

You too can become famous!!

by Matt Giwer

This is to encourage new authors. You spend hours, nay, days or weeks, slaving over a program. Now it works. What do you do with it? Send it to ACE? Definitely. Anything else? Yes, please read on.

This is one person's account, mostly true, of how I got into selling software.

I have had five programs published (two for money) by various sources and five more optioned by COMPUTE! How did I get started? It all goes back to 1967 when I was hired by the Navy and there was little for me to do while waiting for my clearance to come through. The man who was assigned to break me in suggested I learn programming. I allowed as how that was a good way to kill time and in six months I was assigned cognizance of a major computer-aided design effort. [I knew how to program, didn't I?] It is hard to look back upon 1967 and put it in perspective but at the time that must have been a rare skill for an aspiring program manager.

Other than a short period with the SR 52 that was the limit of my hands on experience with digital equipment until I ordered the Bally Home Computer from that Chicago outfit. Well at that point I got in contact with Bob Fabris and his user newsletter, The ARCADIAN. First off I sent in this program for LIFE.IV for free publication. (COMPUTE! has optioned an ATARI version.) The next issue Bob advertised it in the back of the issue saying I would sell it for $5. Bob made an entrepreneur of me. He also taught me something. I sold three copies, not one of them west of the Mississippi. Within six months it was being blatantly offered for sale by other parties on the West Coast without even the courtesy of changing the name. My skull may be thick, but even that got through.

Later Bob worked a deal with ASTROVISION, the new manufacturers of the Bally Arcade, to include an ad for the Newsletter and he offered me $0.25 per subscription he received to include one of my programs. In the summer of 1981 I jumped at found money. In the Fall I bought an ATARI. (Bless you, IRS Rule Makers.)

Frankly I first contacted this group because I thought the ATARI was just like the Bally and Eugene ACE was just another version of the ARCADIAN. Later I discovered there were national magazines which covered the ATARI computer. (There is some story about the dummy who always falls flat on his face but comes up with a $10 bill in his mouth.) So after three months I mailed off a few articles and some of my programs which I wrote -- modified -- created to learn the ATARI computer. LO! and BEHOLD! there comes a check in the mail.

At that point I had earned about $85 and had about $4000.00 sunk into various computer related equipment. So what was I do? That was a challenge. There I was. I had gotten into selling software by mistake. I wrote up a couple of learning type programs and a check came in the mail. Visions of the next VISICALC or PACMAN danced in my head. Obviously I started cranking out computer programs. As long as I am addicted to them personally and spend half of my time writing them then I might as well make a few bucks on the side. Besides the time required to write the article is nothing compared to the time required to get the program working in the first place. (How the hell Ockers does it, I do not know.)

The article probably isn't what Mike had in mind but I hope it starts other programmers to writing. We need more software.

Interfacing your Atari to the Real World

by Marshall S. Dubin
Michigan ACE Newsletter(Apr & May, 82)

Part One: Basic Input:

With the addition of a few parts and some external circuits, your Atari personal computer can send and receive signals from the "real world." These signals can be the monitoring of alarms, themostats, and a variety of digital and analog sources which in turn can control various motors, appliances, lights, etc. The purpose of this article is to discuss a method in which we can monitor up to four input signals from external soucrces in the real world.

For our purposes we will be using the joystick ports which are located on the front panel of the computer. Figure 1 shows a pin connection diagram of a single port. All four ports share this connection pattern.

Note that there are four pins labled PIA. These pins are connected directly to the internal PERIPHERIAL INTERFACE ADAPTER CHIP of the computer. The pins may be formatted for either input or output. We shall discuss them in more detail in future articles. Additional pins are the trigger pin for joystick trigger buttons, two analog input pins, (used for paddle controllers), a +5v pin and a system ground pin. In this example we shall use the TRIGGER pins. Because there are four ports, each port may control a different device connected to the trigger pin of that port.

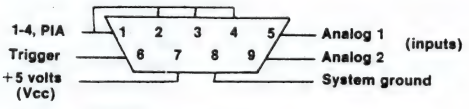


Figure 1: Pin Connections

Basic Input Using the Trigger Pin

When the computer is first switched on, the trigger pins are at logic 1 (high) state. This is the DEFAULT Status of these pins. The normal procedure when using the joysticks is to have the computer react if the pin reads LOW (logic 0). When this is the case, the trigger is assumed to have been pressed. The following listing illustrates this point:

```
10 X=STRIG(0): REM for port one
20 IF X=0 THEN PRINT "TRIGGER IS ACTIVE"
30 PRINT "TRIGGER IS NOT ACTIVE"
40 GOTO 20
```

The computer will take one action when the trigger is pressed and do something else when it is not. Any game using joysticks illustrates this point. Now what if an alarm sensor, liquid level sensor or light activated switch were connected to the joystick trigger input? Ah-ha. We begin to see that the old joystick port can do more than blow away Zylons!

The Basic Input Connection

As I mentioned earlier, the default status of the trigger pin is a logic one. To simulate pressing the trigger button, we must make the status of that pin a logic zero. This is done by connecting that pin to the system ground. When we do that, the pin is said to have been "pulled low," and will now read a zero.

The easiest way to do this is by connecting pin 6 of the controller port to pin 8 of the same port. In the real world situations though, especially if you are monitoring something more than a TTL level signal, this is best accomplished through the use of an external relay or better still an OPTO-ISOLATOR. Lets look at opto-isolators first.

An opto-isolator is composed of a LED (light emitting diode) and a phototransistor. When voltage is applied to the diode, it glows. The light emitted from the diode reacts with the photo sensitive transisor and "biases" it into conduction. Here the transistor acts as a switch which is turned on by the light from the diode, and off when the diode is dark. Since power need only be given to the diode, there is no chance that a higher damaging voltage can cross the optical barrier and damage the computer. As illustrated in figure 2, when the transistor switch is ON, pin 6 of the controller port is connected to pin 8 (ground) thereby pulling it low, and simulating a trigger press.

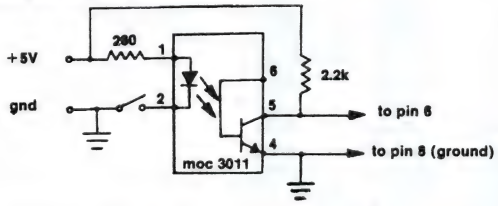


FIGURE 2: Using an opto-isolator

On this example, a relay is powered on by a high voltage such as 110 volts, and this causes the contacts to pull pin 6 low.

Do not try to power the opto-isolator from the "on-board" 5 volt supply. It can only withstand 40 milliamps, and has other uses. Also note that the diodes in some opto-isolators can draw as much as 100 milliamps current, so you will need a supply capable of driving at least one or more of them. In addition, you may have to use a current limiting resisor between the opto-isolator supply and the LED, especially if you use higher voltages to drive the isolator. Use Ohm's law to figure the resistance you will need for the voltage and current ratings you will be using, if they are different from mine.

Figure 3 shows a method of using a relay to pull the trigger pin low. Figure 4 shows a light activated sensor (Sargent and Shoemaker, 1981) which will also work well.

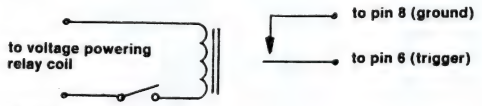


Figure 3: Use of a relay

When switch is closed, a signal is passed to the computer. Switch can be part of a relay if you wish to monitor high current devices. Here your device triggers a relay, which in turn will turn on the opto-isolator.

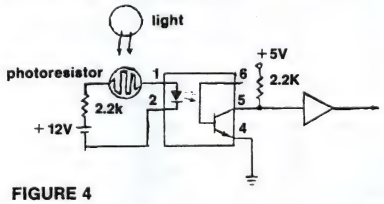


FIGURE 4

The on/off status of a light can be monitored by using a photoresistor as the LED's current limiting resistor. If the photoresistor circuit uses more than five volts, an extra current limiting resistor may be needed. (LED current shouln't exceed 30 ma).

Programming Considerations

The sensors can be used in a program in much the same way as the joystick trigger buttons. The keyword STRIG will read the status of this pin as it does for the normal use of the joystick. Please note that the PTRIG keyword does not access pin 6 of the controller port, but uses different pins. You must use the STRIG keyword, or else PEEK locations 644, 645, 646, or 647 to read the pin status. Each location is for a different port, eg. 644 reads port 1, etc., much the same for STRIG(0).

So Now What

Go to it! Your computer can read and react to all kinds of neat things beside Zylons or Space Invaders. Try using a light beam sensor as a counter, or determing how many times your furnace motor kicks on during the day, or reacting to a metallic "end of tape" sensor for programable slide shows, or even bio-feedback (be SURE to use optical isolation)! What it boils down to is your imagination. Experiment! Learn! Enjoy!

In future articles I will be discussing output as well as the uniqu built in faclity of the Atari to accept analog input without the need for complicated external circuitry. I might also suggest a very good kit made by Mosaic which includes two DB9S plugs, ribbon cable and instructions for interfacing for only $15 (POB 748, Oregon City, Or 97045).

References

Sargent, M and Shoemaker, R. Interfacing Microcomputers to the Real World, Addison-Wesley Publishing Co., Reading, Ma, 1981

Mosaic Electronics Atari I/O Package,, Mosaic Electronis, POB 748, Oregon City, OR 97045

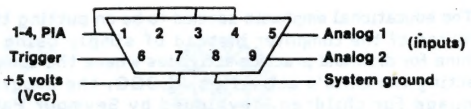
Interfacing your Atari to the Real World

by Marshall S. Dubin
Michigan ACE Newsletter(Apr & May, 82)

Part Two: Controlling Power

We will now look at ways to use the joystick ports for output. This gives you a means to control a wide variety of external devices, relays, and the like.

As you can see from the pin diagram in the figure, the joystick port has several potential input sources available. For example , two of the pins are intended for use with the paddle controllers. These are called the ANALOG pins. They take an analog source such as a variable resistance and convert it into a digial signal. This in essence is how the paddles function. They provide a resistance via a potentiometer within the paddle unit, between the analog input pins and +5 volts DC. The computer interperts the variable voltage as a digial number between 0 and 228. This is called "on board" analog to digital conversion. Units performing a similar function may be purchased at a hefty price, but Atari owners have the use of 8 of these units built right in!



For now, lets concentrate on pins 1-4 on the joystick ports. These are the pins of the Peripheral Interface Adapter chip, commonly referred to as the PIA. The PIA provides a mean of connecting your computer to peripherals, and can be programmed for either input or output. There are two PIA ports of eight bits each available for your use. Joystick ports 1 and 2 compose PIA port A, while joystick ports 3 and 4 compose PIA port B. Each port is one byte (8 bits) and may be used together or individually to provide input and output functions. Some of these functions may be used to drive a printer or other accessory, or even a series of power relays which can control alarms, lights, appliances, motors or whatever.

The problem involved in controlling larger interface devices is a problem of taking a small amount of power and amplifying it. The prots on your computer are not made to power anything more than another chip. the manual recommends a maximum of 1 TTL load (1.6 ma.) for each PIA bit, and no more than 50 ma at the +5v pin. If we are using relays or controlling power, we will need at least 12-24 volts.

There are several ways to accomplish this task. The figures below illustrate some of them. The most common arrangement is the transistor driver. In this arrangement the computer provides a very small voltage which turns on the transistors that in turn switch the load. A second way is through the use of opto-isolators. The computer provides 5 volts which switches the LED of the isolator on. When the diode is lit, this triggers a photosensitive transistor which is connectd to a larger load or relay. A Darlington transistor package can be used in a similar fashion. The low voltage fires the Darlington transistor which can then switch a considerabley larger current. Finally there are integrated circuit interface chips such as the 7407 which allows a switching of up to 30 volts from the 5 volt TTL level of the Atari.

All of these methods will work, and actually which one to use will depend on the specific environment or task lined up for them. The method I prefer is the 7407 chip. It is inexpensive, will handle up to 30 volts and has six gates, so I can switch six devices from a single chip, and will handle about 30ma, which drive most small relays or solid state switches.

The SN7407 made by Texas Instruments is an open collector device. To use it you must connect a 2.2K ohm resistor from each output to +5 volts. When an output is "on" the output is actually open- so the resistor supplies +5 to the device you are driving. You can drive up to 30 volts at the outputs, by adjusting the value of the resistor. When an output is "off", it is shorted to ground, and the device sees 0 volts (ground). The resistor limits this current to a fairly low value so you don't blow the power supply or worse, the chip! Since the resistor can't supply much current, make sure the resistor/7407 combination is seen as the "ground side" of your circuit, i.e., to drive a relay, conect +5v to one side of the relay, and the other side to the output of the 7407. Then, to turn the relay on, turn the 7407 off. Current will flow through the relay, and then through the 7407 to ground.

You can easily drive descrete LED's with this too (such as for test lights), as well as a variety of small relays or solid state switches. Just make sure you sink the current with one end of your driven device to +5v through a resistor and the other end to the 7407. Sending a "0" (logic level low) to the PIA turns it off. If you want to do it the other way around, use the inverting 7406, which will turn your device on with a high logic level and off with a low level. Recognize though that the default state of the PIA when the computer is powered up is all bits high. If you are using an inverting 7406, your devices would come alive when you powered on the Atari. This is way I prefer to use the 7407, since I can power up and then have my software drive the devices by writing a 0 to the bit I want to power a device from.

Speaking of bits, a few words are in order about the structure of the ports before you run off to warm up your soldering irons. The PIA as I mentioned earlier consists of two ports, A and B(PORTA and PORTB). These are controlled through the use of the control registers for each port, PACTL and PBCTL. You may have heard of the PACTL because thats the one you poke with 52 to turn on the cassette player. The addresses are as follows:

PORTA 54016/$D300- port A address
PORTB 54017/$D301- port B address
PACTL 54018/$D302- port A control

☆ ☆ ☆ ☆ ☆ ☆

PBCTL 54019/$D303- port B control

On power up, the ports are initialized to $FFFF (all bits high). To use a port for input, just pull the bit of your choice low by connecting it to ground. To use the port for output, it must be formatted for output. The procedure is not complex:

1. Poke the control register (PACTL or PBCTL) with 56/$38.
2. Poke the port (PORTA or PORTB with 255/$ff. This specifies the port will be used for output.
3. Poke PACTL or PBCTL with 60/$3C.
4. Now just poke the port (PORTA or PORTB with your data.

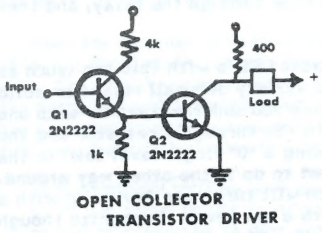
Essentially you have a total of 16 bits to play with. Just remember that two joystick ports make up one PIA port. Stick 0 and 1 are the A side and stick 2 and 3 are side B. Each joystick port is 4 bits or 1 nybble. Each side of the PIA is 8 bits or 1 byte. When programming for output, you must remember that a specific BIT is driving a device. Therefore one joystick port can drive 4 devices (1 for each bit). An entire PIA side will handle 8 devices and if you use both A and B sides you cna trigger 16 individual devices at once or in any combination. You must POKE into that port a decimal number whose BINARY representation will switch on a certain bit or series of bits. For example, if I POKED a 255 into port A, all bits would be on. If I POKED a 12 into port A, bits 3 and 4 only would be on. The individual joystick ports may be read using the shadow registers as follows:

Jack 1 (STICK(0)) 632/$278
Jack 2 (STICK(1)) 633/$279
Jack 3 (STICK(2)) 634/$27A
Jack 4 (STICK(3)) 635?$27B

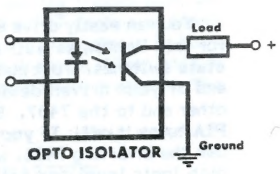
You can also use the BASIC keywords STICK to access these ports, eg. X=STICK(0), etc.

The program listing will provide you with a demonstration on how the ports are prograemed. The program first allows you to select a port, and program it for either input or output. then you can write data to the port and the computer will peek the port and verify the data you wrote. Granted this isn't elaborate, but it workds. Future articles will delve into useful contruction projects, so keep posted.

Reprinted from M.A.C.E. POB 2785, Southfield, MI 48037 ($15 year)



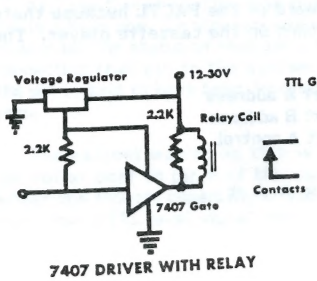
OPEN COLLECTOR TRANSISTOR DRIVER



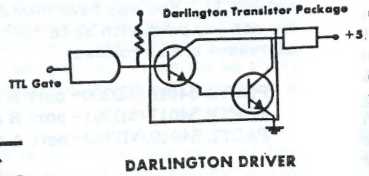
OPTO ISOLATOR

The program listing will provide you with a demonstration on how the ports are programed. The program first allows you to select a port, and program it for either input or output. then you can write data to the port and the computer will peek the port and verify the data you wrote. Granted this isn't elaborate, but it works. Future articles will delve into useful contruction projects, so keep posted.

Reprinted from M.A.C.E. POB 2785, Southfield, MI 48037 ($15 year)



7407 DRIVER WITH RELAY



DARLINGTON DRIVER

```
1 REM REPRINTED FROM M.A.C.E., POB 2785,
 SOUTHFIELD, MI 48037 $15 YR.
10 REM PROGRAM TO FORMAT PIA PORTS
20 REM by Marshall S. Dubin
30 GRAPHICS 0:POSITION 10,2
40 DIM IO$(10),DATA$(3)
50 PRINT "PIA PORT DEMO"
70 REM xxxPORT ADDRESSxxx
90 PORTA=54016:PORTB=54017
100 REM xxxROUTINE TO CONFIGURE PORTxxx
130 TRAP 130:? :? "Configure which port
(1-4)";
140 INPUT PORT:IF PORT<1 OR PORT>4 THEN
130
160 REM xxxSECLECT PORT CONTROL REGISTER
xxx
170 REM xxx  ADDRESS (PACTL, PBCTL)  xxx
190 IF PORT<3 THEN PCTL=54018:PORT=PORTA
200 IF PORT>2 THEN PCTL=54019:PORT=PORTB
210 ? :?
230 REM xxx SELECT INPUT OR OUTPUT  xxx
250 PRINT "Input or Output ";
260 TRAP 250:INPUT IO$
270 IF IO$(1,1)="I" THEN F=0:GOTO 340
280 IF IO$(1,1)="O" THEN F=255:GOTO 340
290 GOTO 250
320 REM xx CONFIGURE THE PORT  xx
340 POKE PCTL,56
350 POKE PORT,F
360 POKE PCTL,60
390 REM xx ENTER YOUR DATA  xx
410 IF IO$(1,1)="I" THEN PRINT "PORT IS
FORMATTED FOR INPUT":PRINT :GOTO 130
420 PRINT "NOW ENTER YOUR DATA"
430 PRINT "(ENTER A RETURN TO DO ANOTHER
 PORT)"
440 INPUT DATA$:IF DATA$="" THEN PRINT C
HR$(125):GOTO 130
450 TRAP 530
470 REM xxPOKE DATA TO PORT/VERIFY ITxx
490 POKE PORT,VAL(DATA$)
500 PRINT "VERIFY ";PEEK(PORT)
510 GOTO 440
520 END
530 TRAP 40000:PRINT "INPUT ERROR, RE-EN
TER ";:GOTO 440
```

The Computer: Extension of the Mind

by Merrianne Coon

Computers are coming to the classroom! The University of Oregon College of Education's decision to devote its 3rd Annual Summer Conference to current and anticipated computer usage in the schools gives credence to this fact.

Noted speakers such as the University's resident expert on computers in education, Dr. David Moursund, Dr. Alfred Bork, respected professor of physics from the University of California at Irvine and luncheon speaker, Harold Kinne, President of Halkin Computing Incorporated from Richardson, Texas presented their personal views of the "electronic revolution." Mr. Kinne pulled both familiar and "state of the art" electronic gadgetry from the inside and outside pockets of his clothing until the audience could not help but be convinced that we were indeed part of an amazing era.

In addition to the major sessions there were numerous special interest sessions and by the end of the conference the message appeared clear. Our schools must adjust to the times if they are to help prepare children to survive in a rapidly changing world. We must take advantage of computers to extend the power of the human mind. Dr. Moursund's analogy that the computer could be an extension of the mind just as a saw and hammer could be an extension of the human body seemed appropriate.

The educational emphasis seemed to be on putting the child in charge of the computer instead of simply using it as a machine for drill and practice activities where the computer is directing the child's activities. LOGO, the programming language for children, developed by Seymour Papert at Massachusetts Institute of Technology was demonstrated and mentioned throughout the conference as a method of using a computer to help a child learn to "think about thinking."

Using inferior or inappropriate software and allowing children to use computers as a buffer to avoid social interactions were among the cautions mentioned concerning the adoption of this new technology in our schools. But the popularity of the conference and the breadth of thoughtful questions seemed to show that educators are also concerned and are willing to take charge of helping direct this potentially revolutionary and powerful "extension of the human mind."

Keyboard for the 400

Several members have added regular keyboards to their Atari 400's, using various methods. When an Atari 400 is opened, and therefore voiding your warrenty, you can easily see where the membrane keyboard is attached to the motherboard through a ribbon cable and a 22-pin connector. A 22 conductor Ribbon connector can then be soldered to the connector, or a 25 pin short stub connector can be attached along with a 25 conductor Ribbon and a 25 pin connector such as a Radio Shack 276-1565 attached to the other end. If you use the 25 pin connector method, you have an outside connector to attach your new keyboard to. The inside connector needs to be a stubby small type because of lack of room.

Various keyboards can be used- if you can get the one used for the Atari 800, interfacing will be the easiest, but many standard keyboards will work. Mount the keyboard in some type of enclosure (a Lee Wards Artist Brushbox will work, or make one from 1/8" plywood), and mount 4 momentarily on, push button switches for the console switches.

To wire your keyboard to your 400, you may need to use some type of connector to connect the keyboard to the ribbon. In general, pin 1 of the keyboard goes to pin 1 of the Atari connector, pin 2 to 2, etc. to pin 17. Pin 18 starts the console switches and goes to the reset switches. Wire one side of the consule swiches to a common ground wire and attach to pin 22 on the Atari, which is the ground. The keyboard wiring diagram, by Denis Beddle of Australia, should help you interface other keyboards.

The above article is based on articles by Nestor Sanchez from the San Diego ACE newsletter (may 82) and from Denis Beddle, an Australian.

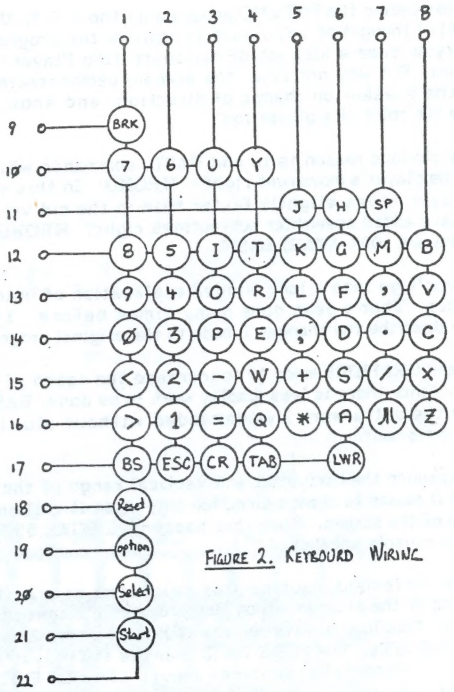
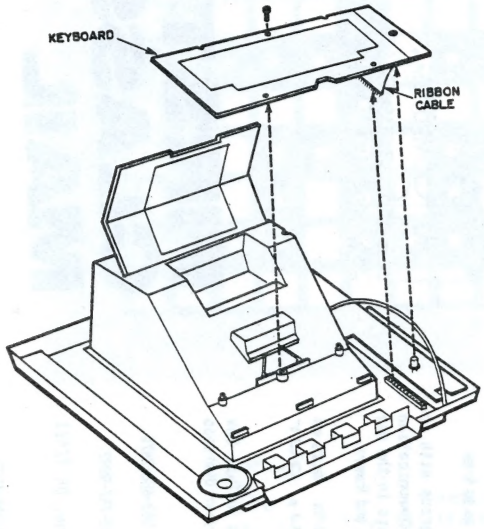


FIGURE 2. KEYBOARD WIRING



Send a business-size SASE to the Ross' for the new, updated ACE Library List!!

Best of ACE-1981, still available, Disk or Tape, $8!

Axlon RAMDISK 128K

Axlon, 170 N. Wolfe Rd., Sunnyvale, CA 94086 (408) 730-0216, $600

Th Ramdisk is a 128K memory board that fits into the middle memory board slot of the Atari 800, increasing your memory to 128K. A 16K board is put in front and back of the board, or Axlon sells a modified 32K board that goes in the first slot if you want the last slot to be empty for other boards such as the BIT-3 80-col board. It comes with software that allows it to be used as a super-fast electronic "disk drive". A special DOS is used, and you load the RAMDISK with the same commands as used for a regular disk drive. After the RAMDISK is loaded, programs can be loaded to and saved from the computer at incredible speed. It works especially good for making multiple copies of disks. The Ramdisk uses the middle of memory for operation, and occasional programs will not run with it because of memory conflicts. For these programs, a little switch on the ramdisk is used to turn it into a regular 16K memory board. There are several programs that will soon be available to utilize the unique features of the Ramdisk, including Filemanager 800 from Synapase, and Fileit 2+ from Swifty.

There are some problems, however. Since you must first boot up the special DOS, any disk that is self-booting and/or protected like VisiCalc, MicroSoft BASIC, etc. can not use the new features. Pascal, a perfect use for it since it requires 2-drives and could use the speed, does not work with it because of memory conficts. I originally bought it to use with our Bulletin Board system, but had some problems. Axlon has been super in trying to help, but there is an apparent conflict between the Ramdisk and our Bulletin Board program under certain condtions. The other problem is the old hair-dryer fuse blowing-anything in memory is lost. Axlon volunteered to refund our money, but we decided it would be perfect for Chuck and Jody Ross and the exchange library to greatly speed up the copying of the many disks they send out. So, at least until we make enough money to buy a double-density drive, we are back to one drive on the bulletin board and not as many programs for you to down-load.

-Mike Dunn

PMDEMO Upgrade (ACEPM) by Jerry White

After seeing the PMDEMO program on the A.C.E. Utilities Disk #1, I thought of a few ways to improve the program. The old version does a nice job of demonstrating Player/Missile Graphics. But why not clear the screen, demonstrate sound, click the speaker on change of direction, and show how to change the color of a player too?

The obvious reason is to add BASIC commands which slow down the player's movement right? WRONG! In this version, the player moves slightly faster than in the old version. I must have added assembler subroutines right? WRONG! Both versions are 100% ATARI BASIC.

I won't go into a long winded explanation of using P/M Graphics. That's been done many times before. I'll just briefly describe the changes I made to the original program.

I used GRAPHICS mode 19 to increase the speed of ATARI BASIC. Since there is less display work to be done, BASIC can handle the extra work I've demanded without slowing the player movement.

I expanded the horizontal and vertical range of the player so that it seems to change direction only when it collides with an edge of the screen. When this happens, I POKE 53279,0 to click the console speaker.

The movement routine was relocated nearer to the beginning of the program which also provides a slight increase in speed. This loop always returns to line 10 where I've added two more POKEs. The POKE 704,Z changes the player's color using its horizontal position as input. The POKE 53760,Y provides the sound using the player's vertical position as input.

Note in line 3000, there is a POKE to location 53761. This combination of POKEs to location 53760 and 53761 replace BASIC's SOUND command. So why didn't I just use a sound command? Once again, for speed.

Location 53760 provides the pitch or frequency for voice zero while location 53761 provides the distortion and volume. The value 130=16*8+2, or distortion level 8 with a volume of 2. As the player reaches the top of the screen, Y=20. The sound you hear at that point is the same as the sound generated by the BASIC command SOUND 0,20,8,2.

To END this program, press the SYSTEM RESET key.

```
1 DIM A$(512),B$(20):X=0:GOTO
110
10 A$(Y,Y+11)=B$:POKE 53248,Z:
Y=Y+V:Z=Z+H:POKE 704,Z:POKE 53
760,Y:IF Y>220 OR Y<20 THEN V=
-V:POKE 53279,0
20 IF Z>205 OR Z<40 THEN H=-H:
POKE 53279,0
30 GOTO 10
100 REM  PLAYER MISSILE EXAMPL
E, FROM        EUGENE ACE PR
OGRAM EXCHANGE
105 REM UPGRADED PMDEMO (D:ACE
PM)    5/25/82 BY JERRY WHITE
110 GRAPHICS 19:TRAP 2000
120 X=X+1:READ A:B$(X,X)=CHR$(
A):GOTO 120
130 DATA 0,24,24,24,24,24,60,6
0,219,219,153,0
2000 POKE 559,62:I=PEEK(106)-1
6:POKE 54279,I:POKE 53277,2
2040 VTAB=PEEK(134)+PEEK(135)*
256
2050 ATAB=PEEK(140)+PEEK(141)*
256
2060 OFFS=I*256+1024-ATAB
2070 HI=INT(OFFS/256):LO=OFFS-
HI*256
2090 POKE VTAB+2,LO:POKE VTAB+
3,HI
3000 Y=60:Z=100:V=1:H=1:SOUND
0,0,0,0:POKE 53761,130:GOTO 10
```